



UNIVERSIDADE ESTADUAL PAULISTA

Pós-Graduação em Engenharia Elétrica

**«ANÁLISE DE ALGORITMOS HEURÍSTICOS PARA O
PROBLEMA DE TRANSPORTE COM CARGA FIXA»**

1210001306



MÁRCIO SIMÕES DE SOUZA

Ilha Solteira - SP



UNIVERSIDADE ESTADUAL PAULISTA - UNESP
FACULDADE DE ENGENHARIA
CAMPUS DE ILHA SOLTEIRA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
CURSO DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

MÁRCIO SIMÕES DE SOUZA

**"ANÁLISE DE ALGORITMOS HEURÍSTICOS PARA O
PROBLEMA DE TRANSPORTE COM CARGA FIXA"**

30400007 ENGA. ELÉTRICA

Doc. 071/2002 - NPD 067/02

UNESP - "CAMPUS DE ILHA SOLTEIRA"	
SERVIÇO TÉCN. DE BIBLIOTECA E DOCUMENTAÇÃO	
DATA DE CHEGADA	DATA DE TOMBO
19.09.02	30.09.02
RECEBIDO POR	TOMBO
Ailza	Te. 1306
AQUISIÇÃO	CLASSIFICAÇÃO
Wesley Autor R\$ 10.00	S729a



Co
Sps - 186283
Sps - 54799

MÁRCIO SIMÕES DE SOUZA

**“ANÁLISE DE ALGORITMOS HEURÍSTICOS PARA O
PROBLEMA DE TRANSPORTE COM CARGA FIXA”**

Dissertação submetida à Faculdade de
Engenharia de Ilha Solteira - UNESP como
parte dos requisitos necessários para
obtenção do título de mestre em Engenharia
Elétrica (ME).

Orientador: Prof.Dr.Rubén Augusto Romero Lázaro

**Ilha Solteira - SP
Fevereiro-2002**

BCpIS - FEIS - UNESP



**ATA DA DEFESA DE DISSERTAÇÃO DE MESTRADO PARA OBTENÇÃO DO
TÍTULO ACADÊMICO DE MESTRE EM ENGENHARIA ELÉTRICA (ME)**

PROGRAMA: ENGENHARIA ELÉTRICA
ÁREA DE CONCENTRAÇÃO: AUTOMAÇÃO
REALIZADA EM: 26 DE FEVEREIRO DE 2002
CANDIDATO: **MÁRCIO SIMÕES DE SOUZA**

COMISSÃO EXAMINADORA:

1. PROF. DR. RUBÉN AUGUSTO ROMERO LÁZARO - (Orientador)
2. PROF. DR. PROF. DR. TAKA AKI OHISHI
3. PROF. DR. JOSÉ ROBERTO SANCHES MANTOVANI

DISSERTAÇÃO INTITULADA: **“ANÁLISE DE ALGORITMOS HEURÍSTICOS
PARA O PROBLEMA DE TRANSPORTE COM
CARGA FIXA”**

LOCAL: Anfiteatro da Biblioteca

HORÁRIO DE INÍCIO: 14 horas

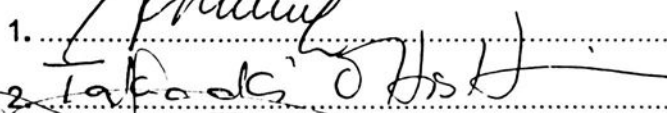
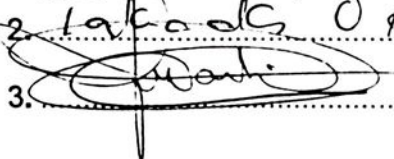
Em sessão pública, após exposição em torno de 40 (quarenta) minutos,
o candidato foi argüido, oralmente, pelos membros componentes da Comissão
Examinadora, tendo demonstrado suficiência de conhecimento e capacidade de
sistematização na pesquisa proposta em sua Dissertação, sendo

Aprovado

Nada mais havendo, foi lavrada a presente Ata a qual é assinada pelos Membros da
Comissão Examinadora.

Ilha Solteira/SP, 26 de fevereiro de 2002.

ORIENTADOR:

1. 
2. 
3. 

“ANÁLISE DE ALGORITMOS HEURÍSTICOS PARA O PROBLEMA DE TRANSPORTE COM CARGA FIXA”

MÁRCIO SIMÕES DE SOUZA

**DISSERTAÇÃO SUBMETIDA À FACULDADE DE ENGENHARIA DO
CÂMPUS DE ILHA SOLTEIRA – UNESP – COMO PARTE DOS
REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO TÍTULO DE
MESTRE EM ENGENHARIA ELÉTRICA (ME).**



Prof. Dr. Antonio Padilha Feltrin – Coordenador

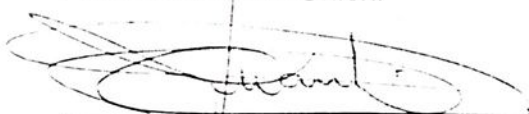
COMISSÃO EXAMINADORA:



Prof. Dr. Rubén Augusto Romero Lázaro - orientador



Prof. Dr. Takaaki Ohishi



Prof. Dr. José Roberto Sanches Mantovani

Ilha Solteira/SP, fevereiro de 2002

AGRADECIMENTOS

Agradeço ao Prof. Rubén pela paciência, pela extrema boa vontade e profissionalismo que teve na orientação desse trabalho, e pela abertura de visão que me proporcionou com relação às possibilidades da Pesquisa Operacional na resolução de problemas em múltiplas áreas de aplicação.



SUMÁRIO

RELAÇÃO DE TABELAS, FIGURAS E ALGORITMOS	IV
RELAÇÃO DE SÍMBOLOS.....	IX
RELAÇÃO DE ABREVIações.....	XII
RESUMO	XIII
ABSTRACT.....	XIV
1. INTRODUÇÃO.....	1
2. PROBLEMAS DE CARGA FIXA	5
2.1 Problema de Localização de Plantas	5
2.2 Problema de Transporte com Carga Fixa	5
2.3 Outros Problemas de Carga Fixa	6
2.4 Abordagens de resolução	7
3. MODELAGEM MATEMÁTICA	9
3.1 Problema de Carga Fixa Geral	9
3.2 Problema de Localização de Plantas	10
3.3 Problema de Transporte	11
3.4 Problema de Transporte com Carga Fixa	14
4. ALGORITMOS HEURÍSTICOS BASEADOS EM PONTOS EXTREMOS PARA O PROBLEMA PTCF	17
4.1 Heurísticas clássicas.....	17
4.2 Heurística de BALINSKI	23
4.3 Heurísticas de WALKER	24
4.4 Heurística de COOPER e DREBES	33

5. ALGORITMO TABU SEARCH GERAL	36
5.1 Memória de curto prazo	41
5.1.1 Manipulação da memória baseada em eventos recentes	41
5.1.2 Critério de aspiração	42
5.1.3 Estratégias para a lista de candidatos	42
5.2 Memória de longo prazo	44
5.2.1 Estratégias de intensificação	44
5.2.2 Estratégias de diversificação	45
5.3 Oscilação estratégica	46
5.4 Reencadeamento de trajetórias (<i>path relinking</i>)	46
6. ALGORITMO TABU SEARCH PARA O PTCF	48
6.1 Conceitos de PTCF e Grafos	48
6.2 Algoritmo TS para o PTCF	50
6.2.1 Obtenção de uma SBF inicial	52
6.2.2 Funcionamento do Algoritmo TS para o PTCF	53
6.2.3 Cálculo da variação da função objetivo	54
6.2.4 Movimento e memórias	56
6.2.5 Soluções vizinhas e avaliação de movimentos.	58
6.2.6 Condições proibidas (tabu)	60
6.2.8 Critério de aspiração	61
6.2.9 Lista de Candidatos	62
6.2.10 Função intensificação, função diversificação e critério de parada	63
6.2.11 Algoritmo TS para o PTCF	66
7. RESULTADOS	75
Problemas Pequenos/Fáceis	77
Problemas Médios	78
Problemas Grandes	79
8. CONCLUSÕES	80
BIBLIOGRAFIA	83
ANEXO A	86
Problemas de Testes e Resultados: Pequenos/Fáceis	86
Problema bk4x3	87
Resolução como PPL por Balinski e Regra do Noroeste	88
Resolução como PPL por Balinski e Regra do Custo Mínimo	89



Resolução pelo Algoritmo TS -----	90
Problema gr4x6-----	91
Resolução como PPL por Balinski e Regra do Noroeste-----	92
Resolução como PPL por Balinski e Regra do Custo Mínimo-----	93
Resolução pelo Algoritmo TS -----	94
Problema puc4x6 -----	95
Resolução como PPL por Balinski e Regra do Noroeste-----	96
Resolução como PPL por Balinski e Regra do Custo Mínimo-----	97
Resolução pelo Algoritmo TS -----	98
Problema murty5x7 -----	99
Resolução como PPL por Balinski e Regra do Noroeste-----	100
Resolução como PPL por Balinski e Regra do Custo Mínimo-----	101
Resolução pelo Algoritmo TS -----	102
Problema bal8x12-----	103
Resolução como PPL por Balinski e Regra do Noroeste-----	104
Resolução como PPL por Balinski e Regra do Custo Mínimo-----	106
Resolução pelo Algoritmo TS -----	108
Problema kb2x3 -----	109
Resolução como PPL por Balinski e Regra do Noroeste-----	110
Resolução como PPL por Balinski e Regra do Custo Mínimo-----	111
Resolução pelo Algoritmo TS -----	112



RELAÇÃO DE TABELAS, FIGURAS E ALGORITMOS

Tabela 3.1 - Problema de Transporte - PT com 3 fábricas(1 e 3 abertas/2 fechada) e 4 consumidores - SALKIN (1975 e 1989) - 13

Tabela 3.2 – Problema de Transporte - Restrições das Ofertas - PT com 3 fábricas(1 e 3 abertas/2 fechada) e 4 consumidores. ---- 13

Tabela 3.3 – Problema de Transporte - Restrições das Demandas - PT com 3 fábricas(1 e 3 abertas/2 fechada) e 4 consumidores. 13

Algoritmo 4.1 - Método Simplex para o PT ----- 20

Algoritmo 4.2 - Regra do Canto Noroeste para o PT ----- 21

Algoritmo 4.3 - Regra do Custo Mínimo para o PT ----- 22

Algoritmo 4.4 - Heurística de BALINSKI para o PTCF ----- 23

Algoritmo 4.5 - Heurística SWIFT-1 de WALKER para o PTCF ----- 30

Algoritmo 4.6 - Heurística SWIFT-2 de WALKER para o PTCF ----- 31

Algoritmo 4.7 - Heurística SWIFT-3 de WALKER para o PTCF ----- 31

Algoritmo 4.8 - Heurística de COOPER e DREBES para o PTCF --- 35

Figura 6.1 – Fluxograma Simplificado do Algoritmo TS para o PTCF ----- 51

Algoritmo 6.1 - Algoritmo TS para o PTCF ----- 66

Tabela 7.1 - Resultados obtidos por Balinski e programação linear – fáceis - bk4x3: problema 4x3 de Balinski e Krarup ap. Gottlieb (2001); gr4x6: problema 4x6 de Gray ap. Gottlieb (2001); puc4x6: problema 4x6 de Puccini (1985); murty5x7: problema 5x7 de Murty (1968); bal8x12: problema 8x12 de Balinski ap. Gottlieb (2001); e kb2x3: problema 2x3 de Kuhn e Baumol ap. Salkin (1989).----- 77

Tabela 7.2 - Resultados comparativos TS e Balinski – fáceis - bk4x3: problema 4x3 de Balinski e Krarup ap. Gottlieb (2001); gr4x6: problema 4x6 de Gray ap. Gottlieb (2001); puc4x6:

problema 4x6 de Puccini (1985); murty5x7: problema 5x7 de Murty (1968); bal8x12: problema 8x12 de Balinski ap. Gottlieb (2001); e kb2x3: problema 2x3 de Kuhn e Baumol ap. Salkin (1989). ----- 77

Tabela 7.3 - Resultados obtidos por Balinski e programação linear – médios - ran10x10a a ran8x32: problemas fctp gerados randomicamente com dimensões 10x10 a 8x32 de Gottlieb (2001). ----- 78

Tabela 7.4 - Resultados comparativos TS e Balinski – médios - ran10x10a a ran8x32: problemas fctp gerados randomicamente com dimensões 10x10 a 8x32 de Gottlieb (2001). ----- 78

Tabela 7.5 - Resultados obtidos por Balinski e programação linear – grandes n3700 a n370e: problemas fctp gerados randomicamente com dimensões 50x100 de Sun et.al ap. Gottlieb (2001). ----- 79

Tabela 7.6 - Resultados comparativos TS e Balinski – grandes - ran10x10a a ran8x32: problemas fctp gerados randomicamente com dimensões 10x10 a 8x32 de Gottlieb (2001). ----- 79



RELAÇÃO DE EQUAÇÕES

Equação 3.1 – PCF - Problema de Carga Fixa----- 9

Equação 3.2 – PPL - Problema de Programação Linear ----- 9

Equação 3.3 – PLP - Problema de Localização de Plantas ----- 10

Equação 3.4 – PLP – Restrição das Ofertas ----- 11

Equação 3.5 – PT – Problema de Transporte----- 11

Equação 3.6 – PTCF – Problema de Transporte com Carga Fixa--- 14

Equação 3.7 – PTCF – Função custo para cada rota (i,j) ----- 15

Equação 3.8 – PTCF – Problema de Transporte com Carga Fixa –
SUN et al. (1998)----- 16

Equação 4.1 – PTCF – SALKIN (1989)----- 18

Equação 4.2 – PT relaxado do PTCF – SALKIN (1989) ----- 18

Equação 4.3 – PTCF – Substituição das variáveis contínuas -
SALKIN(1989) ----- 19

Equação 4.4 – PTCF – Substituição das variáveis binárias - SALKIN
(1989) ----- 19

Equação 4.5 – PTCF – Função Objetivo - SALKIN (1989) ----- 19

Equação 4.6 – Heurística de BALINSKI – matriz de custos
modificados ----- 23

Equação 4.7 – PTCF - Heurísticas de WALKER ----- 25

Equação 4.8 – PTCF - Heurísticas de WALKER ----- 26

Equação 4.9 – Heurísticas de WALKER – critério de decisão da
variável a sair da base ----- 27

Equação 4.10– Heurísticas de WALKER – conjunto das variáveis
que estão na base com valor positivo e assumem valor zero
após mudança da base ----- 28

Equação 4.11 – Heurísticas de WALKER – conjunto das variáveis que estão na base com valor zero e assumem o valor positivo após mudança da base -----	28
Equação 4.12 – Heurísticas de WALKER – variação do valor da função para x_B saindo da base e x_j entrando na base-----	29
Equação 4.13 – Heurísticas de WALKER – máxima variação da função objetivo -----	32
Equação 4.14 – Heurística de COOPER e DREBES - conjunto das restrições-----	33
Equação 4.15 – Heurística de COOPER e DREBES -função objetivo -----	34
Equação 4.16 – Heurística de COOPER e DREBES - matriz de custos modificados -----	35
Equação 6.1 – PTCF – Problema de Transporte com Carga Fixa – SUN et al. (1998)-----	48
Equação 6.2 - Algoritmo TS - Função objetivo para a SBF inicial para o TS -----	52
Equação 6.3 - Algoritmo TS - Variação da função objetivo -----	55
Equação 6.4 - Algoritmo TS - Movimento - 8-upla -----	57
Equação 6.5 - Algoritmo TS - Determinação do arco saindo da base -----	59
Equação 6.6 - Algoritmo TS - Movimento - novo valor de x_{ij} -----	59
Equação 6.7 - Algoritmo TS - Movimento - Mudança líquida da função objetivo -----	59
Equação 6.8 - Algoritmo TS - Movimento - Determinação do melhor movimento -----	60
Equação 6.9 - Algoritmo TS - Condições Tabu -----	61
Equação 6.10 - Algoritmo TS - Critério de Aspiração-----	61

Equação 6.11 - Algoritmo TS - Movimento - Determinação do melhor movimento no processo de memória intermediária -----	63
Equação 6.12 - Algoritmo TS - Cálculo do parâmetro α para o primeiro ótimo local -----	63
Equação 6.13- Algoritmo TS - Movimento - Determinação do melhor movimento no processo de memória de longo prazo 1 -----	64
Equação 6.14 - Algoritmo TS - Movimento - Determinação do melhor movimento no processo de longo prazo 2-----	65



RELAÇÃO DE SÍMBOLOS

α - coeficiente para penalizar ou incentivar configurações a entrar na base, no processo memória intermediária 1 e memória de longo prazo 1, no TS para o PTCF.

β - coeficiente para determinação do número de iterações nos processos memória de curto prazo.no TS para o PTCF.

βN - número de iterações no processo memória de curto prazo e para início do processo memória intermediária 1 no TS para o PTCF.

$(\beta+\gamma)N$ – número limite de iterações no processo memória intermediária 1 e para início do processo memória intermediária 2 no TS para o PTCF.

γ - coeficiente para determinação do número limite de iterações no processo memória intermediária 1 e início do processo memória intermediária 2. no TS para o PTCF.

Δ - variação líquida da função objetivo, correspondente a entrada de (i_e, j_e) na base e a saída de (i_s, j_s) da base, no TS para o PTCF.

Δ' - valor alternativo à variação líquida da função objetivo, correspondente a entrada de (i_e, j_e) na base e a saída de (i_s, j_s) da base, no processo memória intermediária 1 no TS para o PTCF.

Δ'' - valor alternativo à variação líquida da função objetivo, correspondente a entrada de (i_e, j_e) na base e a saída de (i_s, j_s) da base, no processo memória de longo prazo 1 no TS para o PTCF.

Δ_j - variação da função objetivo, nas heurísticas de WALKER para o PTCF.

δ - valor da variável a entrar na base, nas heurísticas de WALKER para o PTCF

θ_j - valor da variável básica x_B ao sair da base nas heurísticas de WALKER para o PTCF.

ξ_{in} – parâmetro correspondente ao tamanho da lista tabu para elementos a entrar na base no TS para o PTCF.

ξ_{out} – parâmetro correspondente ao tamanho da lista tabu para elementos a sair da base do TS para o PTCF.

c_{ij} – custo unitário de transporte da fábrica i para o consumidor j .



$\bar{c}_{ij}$ - custo unitário modificado quando se incorpora ao custo unitário o custo fixo diluído na heurística de Balinski.

$\bar{c}_j$ - custo unitário modificado quando se incorpora ao custo unitário o custo fixo diluído na heurística de Cooper e Drebes.

d_j - demanda do consumidor j .

$f(x)$ - função objetivo, genérica, no TS geral.

f_{ij} - custo fixo ou carga fixa de abertura da rota (i,j) , da fábrica i para o consumidor j .

$g(x_{ij})$ - função custo de transporte na rota (i,j) no PTCF, quando se considera o custo fixo mais o custo variável.

h - memória de histórico da busca no TS geral ou no TS para o PTCF.

k - iteração corrente no TS para o PTCF.

$k_{\min}$ - iteração para a qual se obteve o valor $z_{\min}$, no TS para o PTCF.

m - número de fábricas ou número de linhas das matrizes de custos unitários e custos fixos.

N - número de variáveis de decisão, igual a $m \times n$, no PTCF.

n - número de consumidores ou número de colunas das matrizes de custos unitários e custos fixos.

$N(x)$ ou $N(x_{\text{atual}})$ = lista de candidatos para a configuração x ou x_{atual} do TS geral.

$N^*(x)$ ou $N(h, x_{\text{atual}})$ - vizinhança modificada de x ou x_{atual} do TS geral.

nb - dimensão da base ou árvore geradora mínima, igual a $m + n - 1$, para o PTCF.

s_i - oferta da fábrica i .

t - memória de histórico da busca no TS para o PTCF.

X - conjunto das restrições de um PCF ou PTCF.

x_{atual} - configuração x no TS geral.

$x_{\text{próximo}}$ - vizinho de x no TS geral.

x_o - configuração corrente para a qual foi obtido o valor z_o ou z_c , no TS para o PTCF ou solução ótima local (incumbente) nas heurísticas de WALKER.

x_B = variável básica

x_{best} - configuração para a qual foi obtido o valor z_{best} durante todo o processo no TS para o PTCF.

x_i - valor da variável não básica a entrar na base nas heurísticas de WALKER.

x_{ij} - quantidade transportada da fábrica i para o consumidor j .

x_k - variável não básica, escolhida para se tentar introduzir na base nas heurísticas de WALKER.

$x_{\min}$ - configuração para a qual foi obtido o valor $z_{\min}$ no processo de memória atual no TS para o PTCP.

x_{rs} - valor da variável x_{ij} ao entrar na base no TS para o PTCF.

x_s - configuração ótima local quando $\Delta_j \geq 0$, nas heurísticas de WALKER.

z - matriz do histórico dos valores da função objetivo no TS para o PTCF.

z_{best} - melhor valor da função objetivo durante todo o processo, no TS para o PTCF.

z_k - valor da função objetivo, para a variável não básica x_k , escolhida para se tentar introduzir na base nas heurísticas de WALKER.

$z_{\min}$ - melhor valor da função objetivo durante o processo de memória atual, no TS para o PTCF.

z_o ou z_c - valor da função objetivo para a configuração corrente x_o ou x_c , no TS para o PTCF ou nas heurísticas de WALKER.

z_x - valor da função objetivo para o ótimo x_s nas heurísticas de WALKER.

RELAÇÃO DE ABREVIações

AG - Algoritmo Genético.

AI - Artificial Inteligency ou IA.

B&B - Branch and Bound ou R&L.

BT - Busca Tabu.

FCLP - Fixed Charge Plant Location Problem ou PLP.

FCP - Fixed Charge Problem ou PCF.

FCTP - Fixed Charge Transportation Problem ou PTCF.

GA - Genetic Algorithm ou AG.

IA - Inteligência Artificial

LP - Linear Programing ou PL.

PCF - Problema de Carga Fixa.

PCV - Problema do Caixeiro Viajante.

PLP - Problema de Localização de Plantas.

PPL ou PL - Problema de Programação Linear

PT - Problema de Transporte.

PTCF - Problema de Transporte com Carga Fixa.

R&L - Algoritmo de Ramificação e Limitação.

SA - Simulated Annealing ou SR.

SBF - Solução Básica Factível.

SR - Simulação de Recozimento.

SWIFT - Simplex With Forcing Trials ou Algoritmo Simplex com Tentativas Forçadas.

TP - Transportation Problem ou PT.

TS -Tabu Search ou BT.

TSP - Travelling Salesman Problem ou PCV.



RESUMO

Este trabalho analisa os problemas de carga fixa e particularmente os algoritmos para o problema de transporte com carga fixa aplicados à resolução de problemas teóricos gerados por processos aleatórios dentro de várias faixas de valores de custos unitários e de custos fixos e para várias faixas de ofertas e demandas, para os quais se conheciam os ótimos globais.

Foram analisadas várias heurísticas para a sua resolução e implementadas heurísticas para geração de solução básica factível de partida e para obtenção de um ótimo local utilizando-se uma abordagem de PPL especializado para o problema relaxado em termos dos custos fixos. Foi implementado um algoritmo de Busca Tabu para a sua resolução, utilizando como solução inicial, a solução obtida anteriormente, tendo-se obtido resultados satisfatórios para problemas do tipo 50x100 e resultados bastante satisfatórios, com baixo esforço computacional, para os problemas médios e pequenos.



ABSTRACT

This work analyses the Fixed Charge Problems and particularly, the algorithms in relation to the Fixed Charge Transportation Problems. These algorithms were applied to solving theoretical problems generated by random processes with several ranges of unitary costs values and fix costs for which the optimum global were known.

Several heuristics were analyzed for its resolution and some heuristics were implemented to generate the initial basic feasible solution for the acquisition of a local optimal solution by means of a LP specialized approach for the relaxed problem in terms of its fix costs. A Tabu Search algorithm was implemented for its resolution utilizing as the initial solution that one which was achieved previously, thus reaching satisfactory results in relation to problems type 50x100. The outcome was extremely satisfactory since it resulted in a clear saving of computer effort in relation to solution of medium and small problem.



CAPÍTULO 1

1. INTRODUÇÃO

Um dos problemas que tem tido grande importância nos últimos anos devido à competição global que as empresas vêm sofrendo, seja para forçar a redução de seus custos operacionais, seja para implementar sua estratégia competitiva, é o relativo à logística.

Este problema pode estar relacionado ao fornecimento dos insumos necessários à operação de suas unidades ou se referir à distribuição de mercadorias para pontos de armazenamento ou à entrega de facilidades aos consumidores finais.

Na implementação de uma estratégia *“just in time”*¹ pelo setor industrial ou de serviços, para o fornecimento dos insumos necessários à sua operação, ter uma rede dinâmica, com custos otimizados e que seja facilmente reconfigurável em função de alterações de suas condições, é um aspecto altamente desejável.

Esta questão de logística foi determinante já na 2ª Guerra Mundial, definindo o curso da história, e estava relacionada ao fornecimento de suprimentos necessários ao desenvolvimento das ações no teatro da guerra na Europa.

Hoje, aplicada às empresas, seja na fase do planejamento da rede seja nas operações correntes, a logística é um aspecto de extrema importância para o seu sucesso e sobrevivência, principalmente em países de dimensões continentais como o Brasil.

Pode-se citar duas situações recentes para se destacar a importância da logística: o fornecimento dos suprimentos necessários à escalada Americana no ataque ao Afeganistão onde foram necessárias várias rotas e redes de suprimentos e o problema da geração, transmissão e distribuição de energia elétrica no Brasil, onde se constatou, em determinado período, falta de capacidade de geração ou de “combustível” para as usinas hidroelétricas, que foi agravado pela falta de rotas

¹ Filosofia de manufatura na qual os recursos financeiros, equipamentos e mão-de-obra são alocados somente na quantidade necessária e no momento requerido, procurando se evitar a formação de estoques seja de matérias-primas, seja de produtos acabados, adquirindo ou produzindo os bens somente no exato momento de sua utilização.



(linhas de transmissão) que permitissem utilizar a capacidade excedente em determinadas regiões para atender outras, interligando os vários sistemas.

Quando se envolvem grandes dimensões e um grande número de pontos de suprimento para um grande número de pontos de demanda, levando-se em conta os tipos de custos envolvidos, a minimização dos custos de transportes é um fator decisivo para a competitividade das empresas, mantendo seu posicionamento no mercado através do atendimento de modo eficaz às demandas dos consumidores, representando um aspecto de extrema importância para o sucesso e sobrevivência das empresas.

Um dos problemas de logística que continua tendo grande importância é o da minimização dos custos de transporte quando se tem um grande número de pontos de origem e um grande número de pontos de destino, e ocorrem custos fixos para a implantação das rotas. Este problema passa obrigatoriamente pela utilização de recursos computacionais, seja adotando uma estratégia de solução clássica, seja utilizando uma estratégia heurística.

A tendência atual é a de se procurar obter uma melhor qualidade das soluções dentro de um tempo de resposta adequado, independentemente de qual seja o produto transportado ou do modo de transporte utilizado, seja rodoviário, ferroviário, dutoviário ou mesmo em redes de transmissão de energia e de comunicações por cabos.

O foco desse trabalho é a análise dos problemas de transporte com carga fixa, ou seja, problemas que apresentam custos fixos além dos custos variáveis, considerados como proporcionais às quantidades transportadas, e têm-se como objetivos principais:

i) Analisar, do ponto de vista metodológico, o problema de transporte com carga fixa e seu modelo associado.

ii) Descrever e comparar, genericamente, algoritmos heurísticos para o problema de transporte com carga fixa e implementar computacionalmente um deles para obtenção de uma SBF inicial para o Algoritmo Tabu.

iii) Implementar computacionalmente o Algoritmo Tabu Search baseado em SUN et al. (1998).

iv) Efetuar testes e comparar os resultados obtidos para os problemas de testes de pequenos e simples, de médio porte e de grande porte, utilizando-se os algoritmos implementados computacionalmente.

Assim, efetuou-se a análise, a implementação computacional e testes de algoritmos heurísticos buscando a minimização dos custos de transporte para o PTCF e, particularmente, utilizando-se a abordagem heurística² denominada Tabu Search (TS) ou Busca Tabu³, utilizando-se no trabalho a estruturação abaixo.

No Capítulo 2, é apresentado o problema de Carga Fixa geral e suas variantes correspondentes ao problema de localização de plantas e ao problema de transporte com carga fixa.

No Capítulo 3 são apresentadas modelagens matemáticas dos problemas de carga fixa geral, de problemas de localização e especialmente do problema de transporte com carga fixa, objeto de estudo desse trabalho.

No Capítulo 4 descrevem-se, de modo geral, alguns algoritmos heurísticos desenvolvidos para problemas de carga fixa e para problemas de transporte com carga fixa, sejam para gerar uma solução factível básica de partida para outros algoritmos, heurísticos ou clássicos, sejam para gerar uma solução factível básica inicial de melhor qualidade, que serviram de entrada para o algoritmo Tabu Search desenvolvido.

No Capítulo 5 apresenta-se o procedimento heurístico TS geral, baseado no trabalho original de GLOVER (1996).

No Capítulo 6 destaca-se o procedimento específico para o problema de transporte com carga fixa (PTCF), e para o qual foi implementado o *software* de Busca Tabu, baseado no artigo publicado por SUN et al. (1998:p.441 – 456).

Foi analisada a literatura básica sobre problemas de programação linear geral, sobre problemas de custo mínimo para fluxo em redes e para o problema de transporte. Basicamente foram observados os desenvolvimentos expostos em BAZARAA, JARVIS e SHERALI (1990), HILLIER e LIEBERMAN (1995) e GOLDBARG e LUNA (2000).

No Capítulo 7 são apresentados os resultados obtidos para os problemas de testes considerados pequenos e fáceis obtidos como exemplos na literatura utilizada e para os problemas de testes considerados médios ou grandes, gerados aleatoriamente, de vários tamanhos e níveis de dificuldade, obtidos de GOTTLIEB (2001). Esses problemas são os mesmos utilizados originalmente nos trabalhos de

² Conjunto de regras e métodos que levam à descoberta e à resolução de problemas.

³ Tabu - algo cujo uso é proibido. que não pode ser usado, tocado ou pronunciado. Busca Tabu - algoritmo heurístico que impõe restrições à execução de determinados movimentos.



GOTTLIEB e PAULMANN (1998) e de SUN et al. (1998) e se conhecem seus ótimos globais.

Os problemas foram resolvidos utilizando-se as heurísticas de BALINSKI (1961), fornecendo as soluções factíveis básicas iniciais para o software implementado para o algoritmo TS específico para o Problema de Transporte com Carga Fixa baseado em SUN et al. (1998), sendo seus resultados comparados aos ótimos globais fornecidos para os problemas de teste.

Foi utilizada a linguagem Fortran 77 do Software MS Power Station 4.0 para a codificação dos algoritmos e os testes foram executados num computador Pentium III 550 MHz com 256 Mb de memória RAM.



CAPÍTULO 2

2. PROBLEMAS DE CARGA FIXA

Descrevem-se abaixo importantes exemplos de problemas de carga fixa como os problemas de localização de plantas e os problemas de transporte com carga fixa, que constituem uma variação do modelo geral do problema de carga fixa e abordam-se, de modo geral, as técnicas que têm sido empregadas em sua resolução ao longo dos tempos.

2.1 Problema de Localização de Plantas

No problema de localização de plantas (PLP), tem-se a situação na qual existe um número n de estações de recepção (depósitos ou consumidores) cujas demandas (solicitações) devem ser satisfeitas a partir de m plantas (fábricas) ou depósitos potenciais. Este problema pode ser tratado de modo balanceado, com as capacidades existentes iguais às demandas solicitadas.

Em geral tem-se o número n de consumidores bem maior que o número m de fornecedores e o problema consiste em decidir sobre a melhor localização e sobre a capacidade de cada planta de modo a satisfazer as demandas desses consumidores.

Existe um custo fixo de instalação, chamado de carga fixa, associado à abertura ou construção de cada planta ou depósito, além dos custos variáveis, proporcionais às unidades ou quantidades a serem transportadas. O objetivo do problema PLP é minimizar o custo total, constituído pela somatória dos custos fixos das implantações das plantas, com os custos variáveis decorrentes do transporte das origens implantadas para os destinos, relativamente ao um único produto a ser produzido e transportado.

2.2 Problema de Transporte com Carga Fixa

O problema de transporte com carga fixa (PTCF), objeto de estudo deste trabalho, pode ser derivado do problema anterior quando se tem definido quais as plantas implantadas e passa a se considerar agora a existência de um custo fixo relativo à abertura ou construção de cada rota, além do custo variável proporcional



ao nível de atividade ou quantidade transportada, podendo ocorrer no projeto de redes de comunicação, construção de estradas, oleodutos, gasodutos e cabos óticos, ou ainda no projeto de sistemas logísticos de suprimento ou de distribuição.

O custo total de transporte é, então, a soma do custo fixo de implantação das rotas (custo fixo ou carga fixa) com o custo variável. O custo variável é dependente do nível de produção ou da quantidade transportada e, em geral, dentro de certos limites, proporcional ao nível de atividade (quantidade produzida ou transportada).

Quando ocorrem somente os custos variáveis, ou seja, se os custos de implantação são iguais a zero, é possível utilizar uma formulação de problema de programação linear (**PPL**) para se determinar o ponto ótimo de operação.

Nos casos em que ocorrem custos fixos, deve-se utilizar uma formulação de problema de programação inteira mista (**PPIM**), em que se têm variáveis de decisão inteiras, do tipo 0 - 1 ou sim - não, que correspondem à implantação ou não de uma fábrica ou depósito, e variáveis de decisão contínuas, que são as quantidades a produzir ou a transportar. Particularmente, em alguns problemas, estas variáveis podem ser inteiras.

Para a resolução dos **PTCF** têm sido utilizados softwares especializados para a sua resolução, sejam atuando *on-line* sejam *off-line*, utilizando técnicas derivadas do *branch-and-bound*, porém, quando envolvem grande quantidade de variáveis de decisão, passa-se a utilizar procedimentos heurísticos eficientes como forma de reduzir o custo computacional.

2.3 Outros Problemas de Carga Fixa

Outros tipos de problemas que se enquadram, com adaptações convenientes, aos problemas aqui apresentados são os problemas de localização de facilidades ou de serviços públicos como escolas públicas, hospitais públicos, bases operacionais de polícia e bombeiros. Podem ser enquadrados como problemas de localização de plantas os problemas de construção de usinas termoeletricas ou de subestações de sistemas de energia elétrica. Outro tipo, semelhante aos anteriores, é o problema de designação ou elaboração de escalas de pessoal.

O objetivo, em qualquer situação, é atender as demandas dentro de certos critérios de qualidade desejada, buscando-se obter a otimização dos custos.

De modo similar aos casos relativos à abertura de uma fábrica ou depósito do problema de localização de plantas e aos dos problemas de transporte com carga

fixa, é bastante comum no planejamento e controle da produção, a existência de problemas como por exemplo os dos problemas de alocação de atividades fabris para os centros de trabalho, no qual ocorrem custos fixos de preparação (*setup*), necessários à preparação de cada máquina, para a movimentação dos insumos e para a preparação do pessoal.

O objetivo do problema, além da elaboração da programação da produção propriamente dita, é minimizar os custos totais, levando-se em conta os custos fixos de preparação. Nestes problemas de alocação ou de designação quando se envolve um grande número de tarefas e de máquinas ou pessoas para executá-las, os resultados são obtidas por softwares que empregam atualmente ou procedimentos derivados da técnica *branch-and-bound* ou procedimentos heurísticos eficientes como forma de reduzir o custo computacional devido à ocorrência do fenômeno da explosão combinatorial.

2.4 Abordagens de resolução

Os primeiros algoritmos utilizados na resolução desses tipos de problemas, baseados na técnica de *branch-and-bound*⁴ ou dela derivada, como por exemplo, os algoritmos de planos de cortes, que consistem na introdução de restrições adicionais visando eliminar a cada passo as soluções não-inteiras, até que se obtenha pela programação linear uma solução inteira que seja ótima.

Um outro procedimento bem mais rudimentar que se utilizou foi o de se resolver o problema inicial sem as restrições de integralidade através do método simplex para problemas de programação linear e depois simplesmente arredondar para valores inteiros a solução ótima obtida, o que leva muitas vezes a soluções bastante insatisfatórias, principalmente quando os valores das variáveis de decisão são de pequena magnitude.

Atualmente, devido ao grande desenvolvimento das tecnologias de informação e recursos computacionais disponíveis, empregam-se tanto os algoritmos utilizando métodos aproximados quanto métodos exatos para resolução deste problema.

Utilizando-se a primeira abordagem raramente se encontra o ótimo global mas, em geral, obtêm-se boas soluções, bastante adequadas ao problema, com um

⁴ Algoritmos de busca que utilizam a técnica de dividir e limitar para se sondar a árvore.



pequeno esforço computacional e conseqüentemente num tempo bem menor do que se fossem utilizados algoritmos que buscam encontrar o ótimo exato.

Existe uma grande variedade de algoritmos e uma rica literatura dentro dessa abordagem e que podem ser classificados em algoritmos heurísticos ou metaheurísticos⁵ (*tabu search*, algoritmos genéticos⁶, *simulated annealing*⁷, *greedy*⁸ ou mesmos mistos), e que tiveram um grande desenvolvimento nos anos 80 e 90, aliados a um aumento cada vez maior na velocidade de processamento e na disponibilidade de memórias internas e de massa cada vez maiores e com custos cada vez menores.

Com isso, determinam-se soluções inteiras quase-ótimas, porém bastante satisfatórias quando comparadas às que seriam obtidas por arredondamento da solução contínua do problema relaxado e com um custo computacional bastante inferior aos dos métodos exatos derivados da técnica de *branch-and-bound*.

Em geral, na resolução de problemas de planejamento de redes de transporte, as hipóteses de custos são apenas estimativas, podendo-se nestes casos se optar por uma estratégia de se obter uma boa solução com um custo de computação relativamente baixo, ao invés de se buscar encontrar uma solução ótima porém com um custo de processamento significativamente mais alto. Esta questão deve ser analisada em termos de sua relação custo x benefício, ou seja, analisar o benefício auferido contra o seu custo de obtenção.

Como o valor da demanda é apenas uma projeção estabelecida de modo determinístico para variáveis que têm características probabilísticas e, na prática, os valores das variáveis somente serão conhecidos efetivamente após a implantação do projeto, não sendo exatamente aqueles que foram utilizados nos cálculos preliminares, se reforça a estratégia de se realizar um menor esforço computacional, obtendo-se no entanto uma solução quase-ótima porém de boa qualidade, ao invés de se efetuarem grandes esforços computacionais para se obter um ótimo absoluto porém para dados de entrada que não são exatamente os que serão utilizados no problema real.

⁵ Meta = além. Metaheurística = além da heurística, posterior a heurística.

⁶ Técnica de busca baseada na mecânica da seleção natural e na genética.

⁷ Simulação de recozimento = técnica de busca baseada na mecânica estatística, na qual se simula a elevação da temperatura de um metal, gerando um estado instável provocado pela agitação de suas partículas, e provocando um resfriamento lento até se obter um estado estável.

⁸ Algoritmo guloso = processo de busca onde se busca obter a maior vantagem a cada passo.

CAPÍTULO 3

3. MODELAGEM MATEMÁTICA

3.1 Problema de Carga Fixa Geral

Em geral, qualquer problema de programação inteira mista da forma:

$$(PCF) \quad \min \quad cx + fy \quad (3.1)$$

$$sa \quad x \in X = \{x \mid Ax = b, \quad x \geq 0\}$$

Equação 3.1 – PCF - Problema de Carga Fixa

é chamado de problema de carga fixa (**PCF** ou **FCP - Fixed Charge Problem**), no qual:

$A = (a)$ é a matriz de coeficientes das restrições.

$c = (c)$ é o vetor ou a matriz dos custos unitários.

$f = (f)$ é o vetor ou a matriz dos custos fixos de implantação.

$x = (x)$ é o vetor ou a matriz das variáveis de decisão contínuas correspondentes às quantidades.

$y = (y)$ é o vetor ou a matriz das variáveis de decisão binárias (tipo 1 ou 0) correspondentes às aberturas ou não das atividades.

Observe-se que:

$$y = 1 \text{ se } x > 0 \quad \text{e} \quad y = 0 \text{ se } x = 0.$$

HIRSCH e DANTZIG (1968) ap. SALKIN (1989:p.640) mostraram que uma solução ótima para o problema de carga fixa geral é um ponto extremo da região X e, dentro de certas condições, o problema de carga fixa pode ser resolvido como um problema de programação linear (**PPL**), com a formulação:

$$(PPL) \quad \min \quad cx \quad (3.2)$$

$$sa \quad x \in X$$

Equação 3.2 – PPL - Problema de Programação Linear

A descrição de técnicas heurísticas para o problema de carga fixa geral, nas quais se buscam boas soluções através da análise dos pontos extremos do conjunto X foram feitas, entre outros, por BALINSKI (1961) ap. SUN et al. (1998), COOPER e

DREBES (1967), MURTY (1968), STEINBERG (1970) ap. SUN et al. (1998) e WALKER (1976).

3.2 Problema de Localização de Plantas

Um problema decorrente dos problemas de carga fixa é o problema de localização de plantas capacitado (**PLP ou FCLP- Fixed Charge Capacitated Plant Location Problem**). Ele corresponde ao problema de programação inteira mista, no qual a soma das quantidades demandadas é igual à soma das quantidades produzidas, e se deseja determinar quais serão as plantas i a serem abertas e quais serão as quantidades x_{ij} a serem transportadas de cada planta industrial i aberta para cada consumidor j , de modo a se minimizar o custo total.

Para o problema balanceado o modelo pode ser escrito como:

$$(PLP) \quad \min \quad z = \sum \sum c_{ij} x_{ij} + \sum f_j y_j \quad (3.3)$$

$$sa \quad x_{ij} = d_j \quad (j = 1, \dots, n) \quad (\text{demandas}) \quad (3.3a)$$

$$\sum x_{ij} \leq s_i y_i \quad (i = 1, \dots, m) \quad (\text{ofertas}) \quad (3.3b)$$

$$x_{ij} \geq 0 \quad (\text{para todo } i \text{ e } j) \quad (3.3c)$$

$$y_i = 0 \quad \text{ou} \quad 1 \quad (i = 1, \dots, m) \quad (3.3d)$$

Equação 3.3 – PLP - Problema de Localização de Plantas

Esse modelo representa a situação na qual existem m plantas que produzem um único produto a ser distribuído para n consumidores sendo que, em geral, o número de consumidores é muito maior do que o número de plantas.

O índice i representa uma planta industrial, ou simplesmente fábrica i , que pode produzir até s_i unidades do produto a ser distribuído, e o índice j representa um consumidor que requer d_j unidades do produto distribuído.

Cada f_i é um custo fixo positivo associado com a abertura da fábrica i , enquanto cada c_{ij} representa um custo positivo por unidade transportada da fábrica i para o consumidor j .

As variáveis de decisão são:

x_{ij} = quantidade transportada da fábrica i para o consumidor j

y_i = variável binária igual a 1 se a fábrica i for implantada e igual a 0 se a fábrica i não for implantada.

Se $y_i = 0$, as restrições (3.3b) e (3.3c) implicam que $x_{ij} = 0$ para $j=1,2,\dots,n$, isto é, não teremos produtos transportados de fábricas que não foram abertas.

Se $y_i = 1$, a desigualdade (3.3b) se reduz a:

$$\sum_j x_{ij} \leq s_j \quad (3.4)$$

Equação 3.4 – PLP – Restrição das Ofertas

3.3 Problema de Transporte

Quando se fixa um vetor binário $y = (y_1, y_2, \dots, y_m)$ no problema **PLP (FCLP- Fixed Charge Location Problem)** descrito anteriormente, ou seja, quando se definem quais serão as fábricas a implantar, o problema **PLP** se transforma num **Problema de Transporte (PT ou TP - Transportation Problem)** geral, cujo objetivo é minimizar o custo para suprir as demandas, definindo as quantidades x_{ij} a serem transportadas de cada fábrica i para cada consumidor j .

Este problema pode ser resolvido como um problema de programação linear geral uma vez que ele é um caso particular de problema de programação linear, e pode ser representado pelo modelo matemático abaixo, onde se destaca em separado da função objetivo z o custo fixo total incorrido F , em função das fábricas já terem sido implantadas:

$$(PT) \quad \min \quad z - F = \sum \sum c_{ij} x_{ij} \quad (3.5)$$

$$sa \quad x_{ij} = d_j \quad (j = 1, \dots, n) \quad (\text{demandas}) \quad (3.5a)$$

$$\sum x_{ij} \leq s_i \quad (i = 1, \dots, m) \quad (\text{ofertas}) \quad (3.5b)$$

$$x_{ij} \geq 0 \quad (\text{para todo } i \text{ e } j) \quad (3.5c)$$

Equação 3.5 – PT – Problema de Transporte

O valor de $\mathbf{F}$ é a soma de todos os custos fixos correspondentes às fábricas abertas, ou seja, é a soma dos custos fixos para os quais se tem os $y_i = 1$, sendo s_i as ofertas correspondentes às aberturas dessas mesmas fábricas.

Este problema diz respeito à distribuição de uma única mercadoria (*commodity*) de qualquer tipo, de diversos centros de origem (normalmente fábricas ou depósitos) para vários centros de recebimento (normalmente depósitos ou consumidores), e o objetivo é minimizar o custo total expresso pela fórmula (3.5).

A restrição (3.5a) corresponde às demandas dos consumidores enquanto que a restrição (3.5b) corresponde às limitações de oferta (disponibilidades das fábricas).

No exemplo, pode se considerar que a soma das ofertas disponíveis das fábricas iguala a soma das demandas solicitadas pelos consumidores, tendo-se assim o problema balanceado e cujo modelo têm soluções viáveis.

Se a soma das ofertas for maior que a soma das demandas pode-se considerar um consumidor fictício de modo a se balancear o problema, e ao qual seria destinada a quantidade excedente produzida. De modo semelhante, caso as demandas ultrapassem as ofertas, para se balancear o problema, pode-se considerar um fornecedor fictício ao qual seriam atribuídas as quantidades faltantes e que não seriam entregues.

Um aspecto importante a se observar nos problemas de carga fixa é que, se as ofertas e demandas forem valores inteiros, todo ponto extremo definido pelo conjunto de restrições é inteiro e, portanto, a solução ótima do problema tem os x_{ij} inteiros (para todo i e j). Para o **PT** os pontos extremos ocorrem nas intersecções das restrições impostas e definidas pelas relações (3.5a), (3.5b) e (3.5c).

Para ilustração dos problemas de transporte mostra-se a seguir um exemplo clássico apresentado de forma bastante semelhante nas publicações SALKIN (1975), SALKIN (1989) e HILLIER e LIEBERMAN (1988:125-128), onde se destaca na Tabela 3.5 a sua estrutura matricial particular.

O exemplo consiste num problema de transporte com 3 fábricas e 4 consumidores, sem ocorrência de custos fixos, sendo que HILLIER e LIEBERMAN (1988:125-128) consideram a abertura das 3 fábricas enquanto que SALKIN (1975:482-483) e SALKIN (1989:640-641), consideram que as variáveis binárias assumem os valores $y_1=y_3=1$ e $y_2=0$, ou seja, como se fossem implantadas as fábricas 1 e 3, enquanto que a fábrica 2 não é aberta.

Mostra-se abaixo o segundo caso, com as disponibilidades (ofertas) das fábricas representados originalmente como M_1 , $M_2=0$ e M_3 aqui adequadas à notação utilizada nesse trabalho como s_1 , $s_2=0$ e s_3 , enquanto que as quantidades requeridas pelos consumidores (demandas) são d_1 , d_2 , d_3 e d_4 .

A tabela, característica do problema de transporte, é:

Qtes.transp.de Custos Quantidades	Fábrica 1				Fábrica 2				Fábrica 3			
	C_{11} X_{11}	C_{12} X_{12}	C_{13} X_{13}	C_{14} X_{14}	C_{21} X_{21}	C_{22} X_{22}	C_{23} X_{23}	C_{24} X_{24}	C_{31} X_{31}	C_{32} X_{32}	C_{33} X_{33}	C_{34} X_{34}
Restrição 1	1	1	1	1								
Restrição 2					1	1	1	1				
Restrição 3									1	1	1	1
Restrição 4	1				1				1			
Restrição 5		1				1				1		
Restrição 6			1				1				1	
Restrição 7				1				1				1

Tabela 3.1 - Problema de Transporte - PT com 3 fábricas(1 e 3 abertas/2 fechada) e 4 consumidores - SALKIN (1975 e 1989)

As restrições 1 a 3 são as relativas às ofertas (capacidades das fábricas):

$$C_{11}.X_{11} + C_{12}.X_{12} + C_{13}.X_{13} + C_{14}.X_{14} = S_1.y_1 = S_1$$
$$C_{21}.X_{21} + C_{22}.X_{22} + C_{23}.X_{23} + C_{24}.X_{24} = S_2.y_2 = S_2 = 0$$
$$C_{31}.X_{31} + C_{32}.X_{32} + C_{33}.X_{33} + C_{34}.X_{34} = S_3.y_3 = S_3$$

(cap.da fábr.1)

(cap.da fábr.2)

(cap.da fábr.3)

Tabela 3.2 – Problema de Transporte - Restrições das Ofertas - PT com 3 fábricas(1 e 3 abertas/2 fechada) e 4 consumidores.

As restrições 4 a 7 são relativas às demandas (atender exatamente a solicitação dos consumidores):

$$C_{11}.X_{11} + C_{21}.X_{21} + C_{31}.X_{31} = d_1$$
$$C_{12}.X_{12} + C_{22}.X_{22} + C_{32}.X_{32} = d_2$$
$$C_{13}.X_{13} + C_{23}.X_{23} + C_{33}.X_{33} = d_3$$
$$C_{14}.X_{14} + C_{24}.X_{24} + C_{34}.X_{34} = d_4$$

(demanda do consumidor 1)

(demanda do consumidor 2)

(demanda do consumidor 3)

(demanda do consumidor 4)

Tabela 3.3 – Problema de Transporte - Restrições das Demandas - PT com 3 fábricas(1 e 3 abertas/2 fechada) e 4 consumidores.

3.4 Problema de Transporte com Carga Fixa

No problema de transporte anterior, que é o exemplo clássico, supõe-se que não ocorram custos fixos com a abertura das rotas mas apenas com a abertura das fábricas, cujos custos fixos são tratados em separado uma vez que eles foram fixados preliminarmente quando da abertura das fábricas.

Considerando-se que as fábricas já estão abertas e que dispõem de suas capacidades de produção e ainda que, cada vez que se faz o transporte de uma fábrica i para um consumidor j , além dos custos variáveis que são função (e dentro de certos limites diretamente proporcionais) das quantidades transportadas, tem-se ainda a ocorrência de custos fixos relativos à abertura das rotas, de modo que o problema se transforma num problema de transporte com carga fixa e que constitui um problema de programação inteira mista, devido ao fato das variáveis de decisão serem de dois tipos: inteiras do tipo 0 – 1 e contínuas.

Em geral as variáveis correspondentes às quantidades transportadas (x_{ij}) são contínuas (podendo ser inteiras no caso em que as ofertas e as demandas são inteiras), enquanto que as variáveis (y_{ij}) relativas à existência ou não dos custos fixos devidos à ocorrência ou não de transporte na rota correspondente são binárias.

O modelo matemático para esta situação se escreve como:

$$(PTCF) \quad \min \quad z = \sum \sum (c_{ij}x_{ij} + f_{ij}y_{ij}) \quad (3.6)$$

$$sa \quad x_{ij} = d_j \quad (j = 1, \dots, n) \quad (\text{demandas}) \quad (3.6a)$$

$$\sum x_{ij} \leq s_{ii} \quad (i = 1, \dots, m) \quad (\text{ofertas}) \quad (3.6b)$$

$$x_{ij} \geq 0 \quad (\text{para todo } i \text{ e } j) \quad (3.6c)$$

$$y_{ij} = 0 \quad \text{ou} \quad 1 \quad (\text{para todo } i \text{ e } j) \quad (3.6d)$$

Equação 3.6 – PTCF – Problema de Transporte com Carga Fixa

Este modelo representa a situação na qual existem m plantas que produzem um único produto para n consumidores. Cada planta i produz até s_i unidades e cada consumidor j solicita d_j unidades do produto fabricado. O custo por unidade

transportada de i para j é c_{ij} e o custo fixo que ocorre sempre que existe um transporte de i para j é f_{ij} , decorrente da implantação da rota (i,j) .

A variável x_{ij} corresponde às quantidades a transportar da planta i para o consumidor j , e o custo variável que ocorre em cada rota é calculado como $c_{ij}x_{ij}$. Assim, o custo de transporte, de cada fábrica i para cada consumidor j , pode ser representado pela função $g_{ij}(x_{ij})$ definida como:

$$g_{ij}(x_{ij}) = f_{ij} + c_{ij}x_{ij}, \quad \text{se } x_{ij} \geq 0 \quad (3.7a)$$

ou

$$g_{ij}(x_{ij}) = 0 \quad \text{se } x_{ij} = 0 \quad (3.7b)$$

Equação 3.7 – PTCF – Função custo para cada rota (i,j)

A variável binária y_{ij} será feita igual a **1** sempre que ocorrer um transporte da planta i para o consumidor j e igual a **0** se isto não ocorrer.

Observa-se que se $y_{ij} = 0$, tem-se que $x_{ij} = 0$, e não existirá custo de transporte variável ou fixo da planta i para o consumidor j .

De modo semelhante ao que ocorre no problema de localização de plantas, no problema de transporte com carga fixa uma vez fixada a matriz binária (y_{ij}) das rotas onde ocorrerão transportes, tem-se um problema de transporte clássico, o qual pode ser resolvido pelo método simplex básico ou especializado, devendo se tomar o cuidado de não computar os custos fixos relativos às rotas onde se têm $x_{ij} = 0$.

Uma alternativa de resolução é a utilização de uma técnica enumerativa tipo *branch-and-bound* associada à resolução de subproblemas de programação linear, com todo o esforço computacional que isto requer ou alternativamente por métodos heurísticos.

Pode-se apresentar uma outra forma de modelagem para o **PTCF**, representado pelas Equações 3.6 a 3.6d, conforme apresentada por SUN et al. (1998) para o problema balanceado, na qual ficam implícitas as variáveis de decisão y_{ij} , do tipo 0 ou 1:

(PTCF)
$$\min \sum_{i=1}^m \sum_{j=1}^n g_{ij}(x_{ij}) \quad (3.8)$$

sa
$$\sum_{i=1}^m x_{ij} = d_j \quad j=1,2,\dots,n \text{ (demandas)} \quad (3.8a)$$

$$\sum_{j=1}^n x_{ij} = s_i \quad i=1,2,\dots,m \text{ (ofertas)} \quad (3.8b)$$

$$x_{ij} \geq 0 \quad i=1,2,\dots,m \quad \text{e} \quad j=1,2,\dots,n \quad (3.8c)$$

sendo

$$g_{ij}(x_{ij}) = 0 \quad \text{se} \quad x_{ij} \geq 0 \quad \text{se} \quad x_{ij} = 0 \quad (3.8d)$$

ou

$$g_{ij}(x_{ij}) = c_{ij}x_{ij} + f_{ij} \quad \text{se} \quad x_{ij} = 0 \quad (3.8e)$$

Equação 3.8 – PTCF – Problema de Transporte com Carga Fixa – SUN et al. (1998)

CAPÍTULO 4

4. ALGORITMOS HEURÍSTICOS BASEADOS EM PONTOS EXTREMOS PARA O PROBLEMA PTCF

Apresentam-se abaixo as heurísticas clássicas e as heurísticas de BALINSKI, de WALKER e de COOPER e DREBES, que são baseadas na busca em pontos extremos da região factível.

Estas heurísticas servem de base para a implementação do algoritmo **Tabu Search (TS)** ou **Busca Tabu**, no qual se busca, a partir de uma solução básica factível inicial, obter um ótimo local de boa qualidade para o problema PTCF.

A implementação computacional para se obter uma SBF inicial foi efetuada adaptando-se a heurística de BALINSKI (1961), enquanto que a SBF de partida para este algoritmo foi obtida alternativamente pela regra do canto noroeste ou pela regra do custo mínimo, devido as suas simplicidades e facilidades de implementação computacional, embora existam na literatura outras heurísticas para se obter uma SBF de partida como o método de aproximação de Vogel ou de Russel.

4.1 Heurísticas clássicas

A técnica tradicional para resolução do Problema de Transporte com Carga Fixa (**PTCF**) tem sido a técnica de programação inteira *branch-and-bound* (B&B) na qual as variáveis y_{ij} são utilizadas como variáveis de ramificação, fazendo com que cada variável binária y_{ij} , que pode ser 0 ou 1, seja substituída pela variável contínua y_{ij} com $0 \leq y_{ij} \leq 1$, obtendo-se o problema relaxado e, para cada nó ainda não sondado, seja resolvido um **PPL**.

Uma heurística utilizada é a proposta por SALKIN (1989:p.652-653), na forma do teorema, a partir da sua formulação do problema de transporte com carga fixa:



$$(PTCF) \quad \min \quad \sum_{i=1}^m \sum_{j=1}^n (c_{ij}x_{ij} + f_{ij}y_{ij}) \quad (4.1)$$

$$sa \quad \sum_{i=1}^m x_{ij} = d_j, \quad (j = 1, \dots, n) \quad (4.1a)$$

$$\sum_{j=1}^n x_{ij} = s_i, \quad (i = 1, \dots, m) \quad (4.1b)$$

$$x_{ij} = u_{ij}y_{ij}, \quad \text{para todo } i, j \quad (4.1c)$$

$$x_{ij} \geq 0, \quad \text{para todo } i, j \quad (4.1d)$$

$$y_{ij} = 0 \quad \text{ou} \quad y_{ij} = 1, \quad \text{para todo } i, j \quad (4.1e)$$

sendo

$$u_{ij} = \min(s_i, d_j) \quad (4.1f)$$

Equação 4.1 – PTCF – SALKIN (1989)

Se (x^*_{ij}, y^*_{ij}) é uma solução ótima para o problema de programação linear relaxado do **PTCF**, a restrição $x_{ij} \leq u_{ij}y_{ij}$ satisfaz a condição $x^*_{ij} = u_{ij} \cdot y^*_{ij}$.

A prova desse resultado, por absurdo, segue do fato que todos os f_{ij} são positivos e se $x^*_{ij} < u_{ij} \cdot y^*_{ij}$, y^*_{ij} pode ser reduzido ao valor x^*_{ij}/u_{ij} sem violar qualquer outra restrição. Isto leva a um menor valor para a função objetivo, contradizendo a hipótese de (x^*_{ij}, y^*_{ij}) ser ótimo.

Este teorema é utilizado para simplificar a relaxação do **PTCF** e, em particular, utilizando a restrição $x_{ij} \leq u_{ij}y_{ij}$ de modo estrito, ou seja, fazendo $x_{ij} = u_{ij}y_{ij}$, se pode eliminar y_{ij} pela substituição $y_{ij} = x_{ij}/u_{ij}$, e o problema se reduz a um Problema de Transportes Padrão (**PT**) nas variáveis x_{ij} , conforme abaixo:

$$(PT) \quad \min \quad \sum_{i=1}^m \sum_{j=1}^n (c_{ij} + \frac{f_{ij}}{u_{ij}})x_{ij} \quad (4.2)$$

$$sa \quad \sum_{j=1}^n x_{ij} = d_j, \quad (i = 1, \dots, m) \quad (4.2a)$$

$$\sum_{i=1}^m x_{ij} = s_i, \quad (j = 1, \dots, n) \quad (4.2b)$$

$$x_{ij} \geq 0, \quad \text{para todo } i, j \quad (4.2c)$$

Equação 4.2 – PT relaxado do PTCF – SALKIN (1989)

Se x_{ij}^* é a solução ótima de problema de transporte PT, então x_{ij}^* com $y_{ij}^* = x_{ij}^*/u_{ij}$ é a solução ótima para o problema de programação linear relaxado do PTCF.

Os valores x_{ij} e y_{ij} obtidos por:

$$x_{ij} = x_{ij}^* \quad (4.3)$$

Equação 4.3 – PTCF – Substituição das variáveis contínuas - SALKIN(1989)

$$y_{ij} = 1 \text{ se } x_{ij}^* > 0 \text{ e } y_{ij} = 0 \text{ se } x_{ij}^* = 0 \quad (4.4)$$

Equação 4.4 – PTCF – Substituição das variáveis binárias - SALKIN (1989)

caracterizam a solução factível do **PTCF**, com custo total igual a :

$$z = \sum \sum (c_{ij}x_{ij} + f_{ij}y_{ij}) \quad (4.5)$$

Equação 4.5 – PTCF – Função Objetivo - SALKIN (1989)

BALINSKI (1961) ap. SALKIN (1989:p.652-653) mostrou que em muitos casos esta solução é muito próxima do próprio ótimo para o PTCF e, portanto, pode ser utilizada como um limite superior para diminuir a busca num algoritmo de *branch-and-bound*. Ele apresenta e resolve pelo algoritmo proposto o exemplo do problema de transporte com carga fixa com 4 plantas e 3 consumidores, também utilizado para testes das heurísticas desenvolvidas neste trabalho, na categoria dos problemas pequenos e fáceis.

Um ponto a observar, seja para algoritmos exatos seja para algoritmos heurísticos, é que a obtenção de uma boa solução de partida inicial é condição para melhoria da eficiência da busca e diminuição do esforço computacional.

O algoritmo **TS** implementado tem como base o Método Simplex para Problema de Transporte, adequando-se às condições de existência dos custos fixos nas decisões de se efetuar ou não a troca de uma variável não básica por uma variável básica.

O algoritmo simplex para o problema de transporte consiste em obter-se inicialmente uma base ou árvore geradora mínima com $m+n-1$ valores, dentro de uma matriz x_{ij} de quantidades transportadas, tipo $m \times n$, de modo que não se tenham caminhos fechados ou ciclos, podendo-se ter na base alguns valores iguais a zero,

tendo-se neste caso uma base degenerada, sem prejuízo para a resolução do problema.

Caso ocorra algum ciclo dentro da base ou árvore geradora mínima, um dos arcos básicos poderia ser escrito como combinação linear dos demais, contra a hipótese de que tenhamos $m+n-1$ vetores independentes.

O Método Simplex para o Problema de Transporte (Padrão) consiste em:

Algoritmo 4.1 - Método Simplex para o PT

1 - obter uma SBF de partida (base ou árvore geradora mínima) por algum processo heurístico, descritos a seguir.

2 - calcular os valores $c_{ij} - z_{ij}$ para cada variável não básica x_{ij}

3 - se todos os $c_{ij} - z_{ij} \geq 0$ para as variáveis não básicas então PARAR => ÓTIMO GLOBAL.

4 - se $c_{ij} - z_{ij} < 0$ para algum x_{ij} não básico selecionar um elemento x_{ij} a entrar na base.

5 - encontrar um conjunto de valores básicos que juntamente com o valor não básico x_{ij} constitua um ciclo e selecionar para elemento a sair da base o de menor valor x_{rs} .

6 - fazer a troca de x_{ij} por x_{rs} e atualizar a base

7 - voltar ao passo 2

Transformando-se o **PTCF** num **PT** por modificação da função objetivo e zerando-se os custos fixos e mantendo-se o conjunto das restrições pode se obter a solução básica factível inicial do passo 1 utilizando-se as regras heurísticas clássicas: Regra do Canto Noroeste ou do Canto Superior Esquerdo, Regra do Custo Mínimo, Método de Aproximação de Vogel ou pelo Método de Aproximação de Russel.

Utilizou-se para obtenção da SBF inicial as duas primeiras heurísticas para se obter a solução para o problema relaxado, segundo a heurística de BALINSK (1961), e resolvido por um **PPL** especializado para o problema de transporte, descrito abaixo.

A Regra do Canto Noroeste ou do Canto Superior Esquerdo é a mais simples e rápida de se implementar, necessitando porém aproximadamente o dobro das iterações para se chegar num ótimo global para o problema relaxado (**PT**) que corresponde a um ótimo local para o problema original (**PTCF**), do que a Regra do Custo Mínimo.

Na implementação computacional chegou-se, como esperado, ao mesmo valor da função objetivo para o ótimo global do **PT**, para o problema com os custos modificados, embora fossem diferentes quanto ao valor da função objetivo do **PTCF** devido às diferenças entre os custos fixos para a solução obtida, em virtude desse ótimo poder ocorrer em pontos extremos distintos ao se trabalhar como uma ou outra heurística.

No caso da Regra do Noroeste não se levam em conta os custos e a atribuição inicia-se no canto superior esquerdo, caminhando para a direita ou para baixo, não retornando às linhas ou colunas anteriores que já sofreram atribuição, de modo que ao se chegar ao canto inferior direito têm-se todos os x_{ij} atribuídos num total de $m+n-1$ valores constituindo uma base, mesmo que com alguns valores iguais a zero.

Seguem abaixo as regras utilizadas para a SBF de partida., que constituem a Regra do Canto Noroeste:

Algoritmo 4.2 - Regra do Canto Noroeste para o PT

- 1 - fazer $i=1$ e $j=1$
- 2 - fazer $x_{ij} = \min(s_i, d_j)$ = menor valor entre s_i (oferta) e d_j (demanda)
- 3 - atualizar a oferta e a demanda restante:
 $s_i = s_i - \min(s_i, d_j)$ e $d_j = d_j - \min(s_i, d_j)$.
- 4 - se $s_i > 0$ e $d_j = 0$ fazer $x_{ij} = x_{i,j+1}$ e voltar ao passo 2
 se $s_i = 0$ e $d_j > 0$ fazer $x_{ij} = x_{i+1,j}$ e voltar ao passo 2
 se $i=m$ e $j=n$ então
 PARAR => SBF inicial obtida
 do contrário
 fazer $x_{ij} = x_{i,j+1}$ e voltar ao passo 2
 fim

No caso da implementação da Regra do Custo Mínimo levam-se em conta os custos ao se fazerem as atribuições, havendo porém a necessidade de se marcar as células que ainda não sofreram atribuição através da atribuição inicial $x_{ij}=-1$. Ao longo do processo marcam-se as linhas ou colunas, que já tiveram seus valores esgotados, de duas maneiras: com o próprio valor zero para a oferta e demanda no caso em que ambas se esgotem simultaneamente e levando à próxima atribuição de x_{ij} ser um valor degenerado (igual a zero); com o valor -1 para a oferta ou demanda no caso de ocorrer o esgotamento não simultâneo, eliminando-se as linhas ou as colunas que esgotaram suas ofertas ou demandas e não sofrerão mais atribuições. Garante-se desse modo que o algoritmo não volte a essas filas e garanta a obtenção de uma base com exatamente $m+n-1$ valores, mesmo que tenham ocorrido alguns x_{ij} com valores iguais a zero (base degenerada).

Há necessidade de se utilizar um ponteiro para indicar, na eventualidade de se obter valor 0 para x_{ij} , se o algoritmo está numa linha e vai permanecer na mesma linha ou se está numa coluna e vai permanecer na mesma coluna quando ocorre o esgotamento da oferta e da demanda simultaneamente, rotina que foi implementada através do ponteiro $\text{marc}=0$ para indicar que se está trabalhando na linha e $\text{marc}=1$ para se indicar que se está trabalhando na coluna.

O algoritmo se encerra quando se tiver $\text{nb}=\text{nbmax}$, $s_i=0$ e $d_j=0$.

Outro algoritmo implementado para se gerar uma SBF de partida é o que segue, no caso, implementado para a função custo modificada:

Algoritmo 4.3 - Regra do Custo Mínimo para o PT

- 1 - fazer todos os $x_{ij}=-1$ na matriz de transporte
 - fazer $\text{nbmax}=m+n-1$
 - fazer $\text{nb}=0$
 - fazer $\text{marc}=0$
- 2 - para todos os $x_{ij}=-1$ tais que $s_i \neq -1$ e $d_j \neq -1$
 - a. escolher o menor custo $\bar{c}_{ij}$
 - b. se houver empate, escolher $\bar{c}_{ij}$ aquele para que possuir o maior valor entre os valores de $\min(s_i, d_j)$.
 - c. fazer $x_{ij} = \min(s_i, d_j) = \text{menor valor entre } s_i \text{ (oferta) e } d_j \text{ (demanda)}$
 - d. fazer $\text{nb}=\text{nb}+1$

3 - atualizar oferta e a demanda restante:

$$s_i = s_i - \min(s_i, d_j) \text{ e } d_j = d_j - \min(s_i, d_j)$$

4 - se $s_i > 0$ e $d_j = 0$ fazer $x_{ij} = x_{i,j+1}$, $d_j = -1$ e $\text{marc} = 0$ e voltar ao passo 2

se $s_i = 0$ e $d_j > 0$ fazer $x_{ij} = x_{i+1,j}$, $s_i = -1$ e $\text{marc} = 1$ e voltar ao passo 2

se $\text{nb} < \text{nbmax}$ então

se $\text{marc} = 0$ fazer $x_{ij} = x_{i,j+1}$ e voltar ao passo 2

se $\text{marc} = 1$ fazer $x_{ij} = x_{i+1,j}$ e voltar ao passo 2

ao contrário

PARAR $\Rightarrow$ SBF inicial obtida

fim

4.2 Heurística de BALINSKI

BALINSKI (1961) demonstra que o ótimo global do problema de transporte com carga fixa relaxado e resolvido por programação linear padrão é um limitante inferior para o **PTCF** e pode ser utilizada como uma solução básica factível inicial para os algoritmos heurísticos ou *branch-and-bound*, de modo a torná-los mais eficientes ou diminuir o comprimento da busca.

Basicamente a heurística de BALINSKI (1961) para o problema **PTCF** consiste em substituir as matrizes de custos originais por uma única matriz de custos modificados $\bar{c}_{ij}$ cujos elementos correspondem à soma do custo unitário por unidade transportada com a distribuição do custo fixo de abertura da rota obtida pela divisão desse custo fixo pelo menor valor entre a oferta da fábrica i e a demanda do consumidor j .

Algoritmo 4.4 - Heurística de BALINSKI para o PTCF

1 - Calcular o valor dos custos modificados, obtidos pela soma aos custos unitários de transporte dos custos fixos de abertura das rotas, diluídos em relação ao menor valor obtido entre as capacidades para a fábrica i , s_i , e as demandas do destino j , d_j , pode ser formulado como:

$$\bar{c}_{ij} = c_{ij} + \frac{f_{ij}}{\min(s_i, d_j)} \quad (4.6)$$

Equação 4.6 – Heurística de BALINSKI – matriz de custos modificados

2 - Resolver o problema de transporte padrão obtido, agora sem custos fixos, através de um método de resolução de problema programação linear padrão ou especializado para o problema de transportes. Esses métodos, à exemplo do método simplex, consistem basicamente em se percorrer os pontos extremos do conjunto de restrições efetuando-se a troca de um elemento da base constituída por $m+n-1$ valores por um elemento não básico, até que os custos relativos $\bar{c}_{ij} - z_{ij}$ para todos os elementos não básicos sejam todos maiores ou iguais a zero.

3 - Para a configuração que se obteve o ótimo global obtido para o problema relaxado através da formulação de **PPL** especializado para o problema de transporte, calcular o valor da função objetivo, levando em conta os custos unitários iniciais e seus respectivos custos fixos, nos casos em que a variável é maior que zero, obtendo-se assim um ótimo local para o problema original.

Este procedimento foi utilizado neste trabalho para obter-se a solução factível básica que é um ótimo local do PTCF, necessária para a inicialização do algoritmo TS desenvolvido.

4.3 Heurísticas de WALKER

Um dos algoritmos heurísticos baseados na busca em pontos extremos para o Problema de Carga Fixa disponível na literatura é o desenvolvido por WALKER (1976). Esse algoritmo heurístico foi desenvolvido para resolver o problema de carga fixa geral e portanto também pode ser utilizado para um caso especial do problema de carga fixa como o do problema de transporte com carga fixa, objeto de estudo desta dissertação. A análise abaixo é feita em termos deste problema.

A modelagem do problema de transporte com carga fixa, como já apresentada anteriormente, e adequando-se as notações, trocando a variável de decisão binária y_{ij} utilizada anteriormente por δ_{ij} , pode ser descrita como:



$$(PTCF) \quad \min \quad z = \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij} + \sum_{i=1}^m \sum_{j=1}^n f_{ij} \delta_{ij} \quad (4.7)$$

$$\text{sa} \quad \sum_{i=1}^m x_{ij} = d_j \quad j=1,2,\dots,n \text{ (demandas)} \quad (4.7a)$$

$$\sum_{j=1}^n x_{ij} = s_i \quad i=1,2,\dots,m \text{ (ofertas)} \quad (4.7b)$$

$$x_{ij} \geq 0 \quad i=1,2,\dots,m \text{ e } j=1,2,\dots,n \quad (4.7c)$$

e, adicionalmente, com as condições:

$$\delta_{ij} = 0 \quad \text{se} \quad x_{ij} = 0 \quad (4.7d)$$

ou

$$\delta_{ij} = 1 \quad \text{se} \quad x_{ij} > 0 \quad (4.7e)$$

Equação 4.7 – PTCF - Heurísticas de WALKER

Segundo WALKER (1976), este algoritmo é uma versão melhorada de outros algoritmos heurísticos baseados em pontos extremos adjacentes, como os algoritmos apresentados por STEINBERG (1970), COOPER (1975) e DENZLER (1969), para obtenção de uma solução aproximada para o problema de carga fixa e é extremamente rápido obtendo boas soluções para os problemas de carga fixa.

A estratégia fundamental desse algoritmo consiste na implementação de um processo de otimização de duas fases.

Na fase I, é utilizado o método simplex convencional para encontrar um ponto extremo do conjunto convexo, representado pelas restrições do problema, que é um ótimo local, sendo esta fase chamada fase de PL.

Essa fase tem como única diferença em relação ao método simplex convencional o critério de escolha da variável não básica que deve entrar e da variável básica que deve sair da base: ele leva em conta os custos fixos no cômputo da função objetivo do problema e os casos particulares de pontos extremos degenerados.

Na fase II busca-se encontrar um ponto extremo de melhor qualidade na vizinhança do ponto extremo encontrado na fase I. Esse processo pode ser implementado incorporando na base uma ou várias variáveis não básicas

simultaneamente para se obter um ponto extremo vizinho, mas não necessariamente adjacente ao melhor ponto encontrado na fase I.

Esse algoritmo é chamado por WALKER (1976) de **SWIFT** obtido do inglês: **Simplex With Forcing Trials**, isto é, um algoritmo simplex com tentativas forçadas, utilizando um critério de decisão modificado de mudança de base, tendo apresentado três variações com pequenas diferenças entre eles com relação à implementação da fase II.

A abordagem básica utilizada é :

- (1) obter um ótimo local pela utilização do método simplex com uma modificação da regra de seleção da variável a entrar na solução básica.
- (2) uma vez encontrado um ótimo local, procurar por um ponto extremo melhor, pulando sobre outros pontos adjacentes, para retomar a iteração dois ou três pontos extremos adiante.

Reformulando-se o modelo acima para uma forma mais compacta e equivalente do problema de carga fixa geral, tem-se:

$$\text{(PTCF)} \quad \min \quad z = \sum_{j=1}^n f_j(x_j) \quad (4.8)$$

$$sa \quad Ax = b \quad (4.8a)$$

$$x \geq 0 \quad (4.8b)$$

$$f_j(x_j) = c_j x_j + f_j \delta_j \quad (4.8c)$$

sendo

$$\delta_j = 0 \quad \text{se} \quad x_j = 0 \quad (4.8d)$$

ou

$$\delta_j = 1 \quad \text{se} \quad x_j > 0 \quad (4.8e)$$

Equação 4.8 – PTCF - Heurísticas de WALKER

Para resolução do problema, as relações matemáticas são obtidas utilizando-se como base um quadro simplex com as mesmas características do quadro simplex do PPL. Para um quadro simplex disponível, especificado por uma base B , tem-se x_B representando o vetor das variáveis básicas, f_B representando o vetor dos custos fixos das variáveis básicas, a_j representando a coluna j da matriz A , $y_j = B^{-1}a_j$

representando a coluna a_j atualizada para a base corrente B e $z_j = c_B y_j$ como um coeficiente de custo relativo da variável básica j .

Mais especificamente, as fases I e II podem ser assim descritas:

Fase I

Essa fase é muito semelhante ao método simplex convencional para problemas de PPL com a diferença fundamental no critério de escolha da variável não básica que deve entrar na base: em PPL a variável não básica x_j candidata a entrar na base é uma variável que apresenta um coeficiente de custo relativo, neste trabalho definido como $c_{ij} - z_j < 0$, em oposição a notação utilizada em BAZARAA, JARVIS e SHERALI (1990). Porém, no problema de transporte com carga fixa, esse critério não leva necessariamente a um ponto extremo adjacente de melhor qualidade devido aos custos fixos envolvidos, havendo necessidade de uma modificação no algoritmo para se levarem em conta estes custos.

Com relação à variável que deve sair da base valem as mesmas considerações. Supondo que a variável não básica x_j seja escolhida para entrar na base, no simplex convencional a variável básica x_B que deve sair da base é determinada da seguinte forma:

$$\theta_j = \min_k \left(\frac{x_{Bk}}{y_{kj}}; y_{kj} > 0 \right) = \frac{x_{Br}}{y_{rj}} \quad (4.9)$$

Equação 4.9 – Heurísticas de WALKER – critério de decisão da variável a sair da base

sendo θ_j o valor da variável não básica x_j ao entrar na base.

No PTCF, se $\theta_j > 0$, após uma mudança de base, a função objetivo aumenta em f_j devido a $x_j > 0$ e ao mesmo tempo diminui em f_B porque a variável x_B se torna não básica com valor igual a zero, isto levando-se em conta somente os custos fixos. Adicionalmente, a função objetivo aumenta ou diminui em um valor igual a $\theta_j(c_j - z_j)$, como ocorre no método simplex para resolução de PPL, em que a função objetivo diminui, dependendo do sinal de $(c_j - z_j)$, em função dos custos variáveis.

No caso de ocorrência de pontos extremos degenerados, existem ainda aspectos muito importantes no PTCF que devem ser levados em conta e que normalmente são irrelevantes no método simplex para resolução de problemas de PPL.: quando uma variável não básica x_j é escolhida para entrar na base e existem várias variáveis básicas que são candidatas a sair da base, a troca de base pode levar a um ponto extremo degenerado no qual uma ou várias variáveis básicas assumem o valor zero.

Nesse caso, para o problema PTCF, existe uma diminuição adicional da função objetivo devido à eliminação dos custos fixos de todas as variáveis básicas que assumem o valor zero após a iteração.

Assim, sempre que se passa a um ponto extremo degenerado, deve-se processar o cálculo da diminuição da função objetivo de modo adequado. O oposto também deve ser considerado, ou seja, sempre que se sai de um ponto extremo degenerado, deve-se processar o cálculo do aumento da função objetivo das parcelas correspondentes aos custos fixos das variáveis básicas que mudaram de valor zero para um valor positivo.

Definem-se os seguintes conjuntos:

$$S = \{i \mid \frac{x_{Bi}}{y_{ij}} = \frac{x_{Br}}{y_{rj}}\} \quad (4.10)$$

Equação 4.10– Heurísticas de WALKER – conjunto das variáveis que estão na base com valor positivo e assumem valor zero após mudança da base

e
$$T = \{i \mid x_{Br} = 0, y_{ij} < 0\} \quad (4.11)$$

Equação 4.11 – Heurísticas de WALKER – conjunto das variáveis que estão na base com valor zero e assumem o valor positivo após mudança da base

O conjunto S representa as variáveis básicas que passam para o valor zero juntamente com a variável básica x_B que sai da base, mas que permanecem como variáveis básicas. O conjunto T representa o conjunto de todas as variáveis básicas que se encontram na base com valor zero e com $y_{ij} < 0$ e que, portanto, devem assumir valores positivos com a mudança de base.

Portanto, levando em conta os custos fixos e os problemas com os pontos extremos degenerados, o critério de escolha da variável não básica x_j que deve entrar na base é dado pela relação:

$$\Delta_j = \theta(c_j - z_j) + f_j - f_B - \sum_{i \in S} f_i + \sum_{i \in T} f_i < 0 \quad (4.12)$$

Equação 4.12 – Heurísticas de WALKER – variação do valor da função para x_B saindo da base e x_j entrando na base

sendo Δ_j o valor da diminuição da função objetivo quando a variável não básica x_j entra na base.

Segundo WALKER (1976), os trabalhos de DENZLER (1969) e COOPER (1975) omitem inteiramente essas considerações de degenerescência. Os critérios de escolha do vetor a entrar na base por eles utilizados incluem somente os primeiros três termos de (4.12) enquanto que o trabalho de STEINBERG (1970) também estabelece um critério que omite os dois últimos termos, mas cita que o critério “pode ser modificado ligeiramente na presença de solução degenerada, num modo óbvio” STEINBERG (1970) ap. WALKER (1976:589).

A relação (4.12) pode ser utilizada como critério de escolha de uma variável não básica a entrar na base até que se tenha $\Delta_j \geq 0$ para todas as variáveis não básicas. Nesse caso foi identificado apenas um ótimo local para o PTCF. No entanto, em virtude da função objetivo ser côncava, não é verdadeiro que se tenha um mínimo global quando todos os $\Delta_j \geq 0$. Para uma função côncava com restrições em uma região convexa, mesmo que não existam pontos extremos adjacentes que produzam um menor valor para a função objetivo, ainda pode ser possível encontrar alguns outros pontos da região convexa que sejam melhores que o ótimo atual KUHN e BAUMOL (1962) ap. WALKER (1976:589).

WALKER (1976) apresenta uma proposta para estas dificuldades através da busca por um ponto extremo de melhor qualidade que o ótimo local obtido na fase I, pesquisando pontos extremos não adjacentes e de melhor qualidade que o atual.

A proposta de busca local é descrita abaixo:

Fase II

Foram apresentadas três versões ligeiramente diferentes com relação à Fase II, chamadas SWIFT-1, SWIFT-2 e SWIFT-3. Dada à possibilidade da fase I encerrar a busca num ótimo local, para se analisar melhor o espaço de soluções, há necessidade de se implementar a segunda etapa, denominada fase II, na qual uma

ou mais variáveis não básicas são introduzidas na base, aumentando desse modo o valor da função objetivo, mas buscando encontrar um ponto extremo não adjacente ao ponto extremo corrente, que permita passar para um ponto extremo de melhor qualidade do que o ótimo local anterior, ou seja, busca-se encontrar um ótimo de melhor qualidade na vizinhança do ótimo local encontrado na fase anterior.

Ao final da Fase I tem-se $\Delta_j \geq 0$ para todas as variáveis não básicas x_j , podendo esta solução ser ou não o ótimo global. Na Fase II um ou mais vetores serão forçados a entrar na base, aumentando o valor da função objetivo, em virtude da possibilidade de ao continuar as iterações, a partir de um novo ponto, possa levar a busca para um novo ótimo distante do antigo ótimo local, para então melhorá-lo. Isto é, de um ótimo local, é feita uma investigação nos pontos extremos adjacentes com maior valor da função objetivo que os valores da menor função objetivo.

As três versões apresentadas para a Fase II diferem em dois aspectos fundamentais:

- (1) no número de vetores forçados a entrar na base em cada tentativa.
- (2) na decisão a ser tomada se uma tentativa de mudança de base levar a um extremo de pior qualidade.

Apresenta-se abaixo o algoritmo denominado SWIFT-1 que corresponde a ser forçar a entrada de um único vetor na base, sem retorno:

Algoritmo 4.5 - Heurística SWIFT-1 de WALKER para o PTCF

0. Determinar uma SBF para o PTCF
1. Realizar iterações simplex utilizando o critério (4.12) para escolher a variável não básica a entrar na base, até que $\Delta_j \geq 0$ para todas as variáveis não básicas. .
 - a. Seja x_0 a solução encontrada na Fase I.
 - b. Seja z_0 a função objetivo correspondente a x_0 .
2. Experimentar a introdução de uma variável não básica x_k , ainda não analisada, na base, determinando uma nova solução x_k com função objetivo $z_k \geq z_0$. Se todas as variáveis não básicas para a solução atual x_0 já foram experimentadas sem que se obtenha uma solução de melhor qualidade que x_0 , PARE e identifique x_0 como a melhor solução encontrada. Ao contrário, ir para o Passo 3.

3. Para x_k obtido no passo 2, realizar iteração como no passo 1, até que $\Delta_j \geq 0$ para todas as variáveis não básicas x_j e identificar x_s como um ótimo local.
 - a. se $x_s = x_k$, isto é, não foi possível realizar iterações, retornar à solução x_0 e ir ao passo 2.
 - b. Se $z_s < z_0$, então, foi encontrada uma solução de melhor qualidade. Fazer $x_0 = x_s$. Ir ao passo 2.
 - c. Se $z_s \geq z_0$, ir ao passo 2.

O algoritmo SWIFT-2, descrito a seguir, corresponde a se forçar um único vetor a entrar na base, com retorno.

Algoritmo 4.6 - Heurística SWIFT-2 de WALKER para o PTCF

Este algoritmo é idêntico ao SWIFT-1, exceto pela mudança do Passo 3c que passa a ser o seguinte:

3. c. Se $z_s \geq z_0$, retornar à solução x_0 (incumbente).
Ir para o Passo 2.

O algoritmo SWIFT-3, descrito a seguir, corresponde a se forçar dois vetores a entrar na base, simultaneamente, com retorno.

Algoritmo 4.7 - Heurística SWIFT-3 de WALKER para o PTCF

Este algoritmo é idêntico ao SWIFT-2, exceto pela mudança do Passo 2, que passa a ser o seguinte:

2. Introduzir um par de pontos de variáveis não básicas, ainda não analisados, na solução x_0 , obtendo-se uma nova solução x_1 com função objetivo $z_k \geq z_0$.

Se todos os pares de variáveis não básicas para a solução x_0 já foram experimentados sem melhoria da função objetivo, PARE e identifique x_0 como solução (incumbente) ;

Do contrário, ir para o Passo 3.

Refinamento Adicional: Máxima Variação da Função Objetivo.

É possível introduzir-se uma modificação adicional no algoritmo de WALKER de modo que, de modo diferente ao descrito anteriormente, ao invés de se selecionar para entrar na base a primeira variável não básica que produz uma diminuição no valor da função objetivo, adotar-se como critério de escolha para entrar na base a variável não básica que produza a maior diminuição Δ_j na função objetivo.

Desse modo, pode-se trocar o critério utilizado para se determinar a variável não básica x_j a entrar na base do passo 1 pelo critério abaixo, que provoca uma diminuição mais abrupta do valor da função objetivo, dado pelo valor Δ_j , calculado da seguinte forma:

$$\Delta_j = \min\{\Delta_i; i \mid x_i \notin B\} \quad (4.13)$$

Equação 4.13 – Heurísticas de WALKER – máxima variação da função objetivo

Este critério geralmente não é utilizado em problema de PPL devido a ter que se encontrar um valor θ_j para cada variável não básica x_j , que tenha coeficiente de custo relativo positivo, gerando um esforço de processamento adicional, porém, para o caso do PTCF, isso é geralmente adequado.

Na implementação do algoritmo TS propriamente, baseado em SUN et al.(1988), como a escolha da variável não básica a entrar na base é efetuada linha-a-linha, e não em toda a matriz, adotou-se o critério anterior para a escolha na linha, levando a uma convergência mais rápida da busca local ou processo memória de curto prazo e para o processo de memória intermediária 2 onde se parte da solução corrente após ter ocorrido um certo número de iterações, buscando introduzir na base, uma a uma, todas as variáveis não básicas que provocam uma redução da função objetivo.

Segundo WALKER (1976) a implementação do critério anterior para o algoritmo SWIFT provoca um aumento do tempo de cálculo por iteração porém reduzindo em torno de 15% a 30% o número de iterações por problema.

4.4 Heurística de COOPER e DREBES

O algoritmo heurístico de COOPER e DREBES (1967) foi proposto originalmente para resolução do problema de carga fixa geral, podendo no entanto, ser utilizado para um caso especial de problema de carga fixa, como o do problema de transporte de carga fixa, analisado neste trabalho.

Para esta análise podemos utilizar as mesmas formulações propostas por WALKER, nas relações (4.7) para o modelo detalhado e (4.8) para o modelo simplificado.

Segundo COOPER e DREBES (1967), no problema de carga fixa geral e, em particular, para o problema de transporte com carga fixa, tem-se uma propriedade fundamental que especifica que, se o problema tem um ótimo finito, então uma solução ótima do problema é um ponto extremo do conjunto convexo definido pelas restrições do problema, isto é, o ótimo do problema é um ponto extremo da região:

$$X = \{x \mid Ax = b; \geq 0\} \quad (4.14)$$

Equação 4.14 – Heurística de COOPER e DREBES - conjunto das restrições

Esta propriedade é a mesma propriedade empregada em problemas de programação linear (PPL) e que representa a estratégia fundamental do método simplex para resolver problemas de PPL. A diferença fundamental entre um problema de PPL e um problema de transporte com carga fixa é que um problema de PPL apresenta um ponto extremo que é um ótimo global enquanto que num problema de transporte com carga fixa existem muitos pontos extremos que são ótimos locais.

Em ambos, a região factível é um conjunto convexo, porém, em problemas de PPL a função objetivo também é convexa, enquanto que no problema de transporte com carga fixa tem-se uma função objetivo côncava, devido às descontinuidades provocadas pelos custos fixos. Isto leva o problema a ter muitos pontos extremos que são soluções ótimas locais, tornando inviável a aplicação direta do método simplex para o problema de transporte com carga fixa (PTCF).

A idéia fundamental do algoritmo de COOPER e DREBES consiste em se analisar um subconjunto dos pontos extremos da região factível tendo como

estratégia básica à utilização do método de pontos extremos adjacentes modificados.

Em PPL dois pontos extremos são adjacentes quando existe apenas um elemento diferente nas variáveis básicas que representam a SBF. O método simplex trabalha passando de um ponto extremo para um ponto extremo vizinho ou adjacente de melhor qualidade, terminando quando todos os pontos extremos são de pior qualidade, caracterizando este ponto extremo como o ótimo global do problema de PPL. Porém, no caso do PTCF, essa estratégia de pontos extremos adjacentes, para ser empregada, necessita de ser modificada de modo a evitar que o processo pare em um ponto extremo que seja somente ótimo local.

As seguintes propriedades devem ser lembradas:

1. A função objetivo da relação (4.8) é uma função côncava para $x_j \geq 0$.

COOPER e DREBES (1967) apresentam em seu artigo a prova formal de que a função da relação (4.8) é uma função côncava.

2. A soma de funções côncavas $f_j(x_j)$ sobre um conjunto convexo X é uma função côncava e, portanto, a função objetivo do PTCF:

$$z = \sum_{j=1}^n f_j(x_j) \quad (4.15)$$

Equação 4.15 – Heurística de COOPER e DREBES -função objetivo

é uma função côncava sobre o conjunto convexo X .

3. Para conjuntos fechados X , limitados por baixo, o mínimo global de uma função côncava está representado por um ou mais pontos extremos de X .

As propriedades anteriores mostram que a busca do ótimo global para o PTCF pode se restringir aos pontos extremos de X , porém, esta propriedade não é de grande valor como no caso de problemas de PPL. No caso do PTCF o número de pontos extremos é muito elevado (o que também ocorre em PPL) e, mais importante, existem muitos pontos extremos que são ótimos locais, inviabilizando a utilização do método de pontos extremos com base no método simplex. Cada ponto extremo do PTCF é um ótimo local para uma vizinhança definida por um raio pequeno $\rho > 0$. Nesse caso, qualquer ponto da vizinhança é obtido transformando o valor de uma variável não básica $x_j = 0$ para $x_j > 0$ e adicionando à função objetivo o valor f_j , correspondente ao custo fixo correspondente a ela.

O algoritmo proposto por COOPER e DREBES consiste fundamentalmente em se utilizar as estratégias abaixo:

Algoritmo 4.8 - Heurística de COOPER e DREBES para o PTCF

1. Mudar os critérios de troca de base entre a variável que entra na base a variável que deve sair da base, utilizando coeficientes de custo recalculados conforme abaixo:

$$\bar{c}_j = c_j + \frac{f_j}{x_j} \quad \text{para} \quad x_j > 0 \quad (4.16)$$

Equação 4.16 – Heurística de COOPER e DREBES - matriz de custos modificados

e utilizar esses coeficientes de custos modificados como se fossem coeficientes de custos de um PPL.

2. Sob determinadas condições, escolher para variável a entrar na base a variável não básica com o menor valor de custo fixo.

3. Em outras condições, escolher para variável a sair da base a variável com o maior valor de custo fixo.

A entrada na base deve ser feita conforme as regras do método simplex.

COOPER e DREBES (1967) mostram que utilizando essas regras não se garante uma convergência para um ótimo global, entretanto, na prática, segundo eles, o algoritmo tem apresentado sucesso.

A heurística de COOPER e DREBES se divide em várias fases que se subdividem em várias etapas e foi aplicada, em seu artigo, para resolver problemas de diversas ordens gerados aleatoriamente, não tendo sido implementado computacionalmente nesta dissertação.

CAPÍTULO 5

5. ALGORITMO TABU SEARCH GERAL

Etimologicamente, entende-se por tabu algo que está sujeito a influências históricas e culturais e que, em certas condições, podem ser ignoradas. Segundo HOUAISS (2001:2654), no seu verbete 9 diz que tabu s.m. significa algo que não pode ser usado, feito, tocado ou pronunciado, por crença, respeito ou pudor No contexto deste trabalho, entende-se como tabu, as condições que não devem ser utilizadas mas que, se atenderem certos critérios, serão utilizadas para se efetuar um movimento, procurando contornar certas situações, mesmo que se tenha de prejudicar inicialmente o valor da função objetivo.

A BUSCA TABU ou TABU SEARCH (TS) pode ser definida como um procedimento metaheurístico adaptativo, desenvolvido por GLOVER (1996), utilizado para resolução de problemas de otimização combinatorial, que procura efetuar a busca de uma solução quase-ótima, armazenando informações de movimentos passados para melhorar a eficiência da busca e de modo a evitar que o processo pare num ótimo local qualquer. Para isso, utiliza estruturas de memória flexíveis (lista tabu) para manter informações (acumular conhecimento) sobre a história seletiva das soluções encontradas durante a busca. Pelas informações armazenadas das soluções visitadas mais recentemente ou mais freqüentemente (ou menos freqüentemente) monta-se uma tabela tabu que é utilizada para desencorajar a seleção de certas configurações durante o processo de busca, dirigindo assim a busca para outras regiões, fora dos ótimos locais.

Esse algoritmo é considerado relativamente novo dentro das heurísticas inteligentes, tendo apresentado um grande desenvolvimento e um grande número de artigos publicados nos anos 80-90. Atualmente, Fred Glover e Manuel Laguna, professores da Universidade do Colorado, têm sido seus principais pesquisadores..

Ele tem sido utilizado para resolução de uma grande variedade de problemas pesados de otimização, como problemas de alocação, problema do caixeiro viajante (Travelling Salesman Problem-TSP), de roteamento de veículos e, mais recentemente, para resolução de problemas de projeto de redes, e problema de transporte com carga fixa mas, segundo GALLEGU RENDÓN (1997:110-160), a

aplicação de TS em problemas de sistemas de energia elétrica ainda se encontra na fase inicial, tendo sido utilizada eventualmente em problemas de planejamento de sistemas de energia elétrica.

A descrição apresentada abaixo consiste numa tradução e adaptação do texto *Búsqueda Tabú* de GLOVER in: DIAS (Coord.) et al. (1996;105-142). O TS está baseado na premissa de que um algoritmo para ser chamado de inteligente na busca da solução de um problema deve incorporar memória adaptativa e exploração sensível em contraposição a outras heurísticas que não fazem uso de memória como o *Simulated Annealing* (SA), como o Algoritmo Genético (GA) ou que utilizam memória rígida como os *branch-and-bound* (B&B), ou mesmo os baseados em Inteligência Artificial (AI).

O TS, ao contrário do SA e do GA, baseia-se em movimentos determinísticos e não aleatórios, fazendo uso de estruturas especiais de memória e de estratégias de busca dinâmica podendo, eventualmente, ser implementado de modo que a busca possa ser probabilística. O TS está baseado na suposição de que uma má escolha estratégica pode produzir mais informação do que uma boa escolha ao acaso. Num sistema que emprega memória, uma má escolha baseada em uma estratégia, pode fornecer informações úteis a respeito de como poderiam ser feitas modificações proveitosas para esta estratégia ao longo da busca.

O TS procura determinar um mínimo local e, para evitar o retorno aos pontos já pesquisados, armazena os movimentos mais recentes efetuados através de uma lista de Movimentos Tabu ou Proibidos, seja de forma explícita, seja de forma implícita, sendo que esta regra, dependendo da implementação, pode se alterar ao longo da execução do algoritmo.

Inicialmente, procura-se fazer uma pesquisa apurada no espaço de soluções, através do processo chamado de diversificação mas, com a localização dos candidatos, a busca passa a ser mais focalizada, de modo a produzir ótimos locais através do processo chamado de intensificação.

As principais diferenças entre as várias implementações do TS tem sido o tamanho, a variabilidade e adaptabilidade da memória de movimentos proibidos em função do problema particular que está em análise.



Em geral, um procedimento TS básico, consiste de:

Passo 1 – Inicialização

Iniciar com uma SBF e denomine-a x_{atual} e com a memória da história da busca h vazia.

Passo 2 – Escolha

Determinar uma lista de candidatos $N(x_{\text{atual}})$ como um subconjunto de $N(h, x_{\text{atual}})$ e selecionar um valor $x_{\text{próximo}}$ como um candidato a minimizar a função objetivo.

Passo 3 – Atualização

Atualizar a solução atual e a memória h .

Os principais elementos característicos utilizados pelo TS são:

Memória adaptativa

Seletividade (inclusive esquecimento estratégico)

Abstração e decomposição (uso de memória explícita ou por atributos)

Tempo

Eventos recentes

Eventos freqüentes

Diferenciação entre curto e longo prazo

Qualidade e Impacto

Atração relativa de escolhas alternativas

Amplitude das mudanças em relações de estrutura ou restrições

Contexto

Interdependência regional

Interdependência estrutural

Interdependência seqüencial

Exploração sensível

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

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

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

Padrões de busca não monótonos (oscilação estratégica)

Integração e extensão das soluções (*path relinking*) - (reencadeamento de trajetórias)

O TS tem sido uma área de investigação teórica e prática pela incorporação de novas estratégias ou adaptando-se ou mesclando as listadas anteriormente.

Descreve-se a seguir, de modo mais detalhado, o algoritmo de busca tabu básico, para o problema de minimização geral:

Dada uma função $f(x)$ a ser minimizada em um conjunto de restrições X , utilizam-se procedimentos semelhantes aqueles dos utilizados pelos algoritmos de busca local, procedendo iterativamente à busca, de uma solução para outra, até que satisfaça um critério de finalização.

Assim, para cada configuração x , pode-se definir uma vizinhança associada $N(x)$ contida em um conjunto de restrições X , e considera-se como vizinho qualquer configuração x' , pertencente a $N(x)$, que possa ser atingida a partir de x , mediante uma operação chamada de movimento.

O TS consegue evitar a parada num ótimo local, indo além da busca local, através de uma estratégia de modificação da vizinhança $N(x)$ ao longo da busca, substituindo a vizinhança $N(x)$ por outra vizinhança $N^*(x)$. Isto é feito pelo emprego de estruturas especiais de memória que servem para determinar $N^*(x)$ e assim organizar a maneira pela qual se explora o espaço de soluções.

O que diferencia o TS de um algoritmo de busca local são os seguintes aspectos:

- a partir de uma configuração (solução) corrente, passa-se à melhor configuração vizinha ou para a menos pior, sendo permitido uma degradação da qualidade da função objetivo.

- o conjunto de vizinhos de x não é caracterizado de maneira estática mas sim de forma dinâmica, definindo-se uma nova estrutura de vizinhança $N^*(x)$, que varia em estrutura e tamanho durante todo o processo de otimização, permitindo assim ao TS realizar uma busca eficiente e inteligente.

$N(x)$ é definido como o conjunto das configurações candidatas definidas pelo critério ou a estrutura de vizinhança e $N^*(x)$ é o conjunto dos elementos que não estão proibidos pela lista tabu, sendo esse um subconjunto do subconjunto $N^{**}(x)$ reduzido de $N(x)$ por algum critério de redução das configurações.

Uma das maneiras de se determinar quais as soluções que são admitidas em $N^*(x)$ é a identificação de soluções encontradas sobre um horizonte especificado e proibi-las de pertencer a $N^*(x)$ classificando-as como proibidas ou tabu.

Os processos pelos quais as soluções adquirem um estado proibido podem decorrer de vários motivos, como por exemplo:

1 - Para promover um exame mais agressivo e dirigido para novos pontos. Uma maneira de se implementar estas condições é substituir a função avaliação original através de penalidades para impedir de modo significativo a escolha de soluções que são tabus, ou seja, aquelas que serão preferencialmente excluídas da vizinhança $N^*(x)$, de acordo com sua dependência dos elementos que definem o estado tabu.

2 - As avaliações tabus incluem também, de modo periódico, incentivo para estimular a escolha de outros tipos de soluções, como resultado do critério nível de aspiração e influências de longo prazo.

Para se implementarem estas funções o algoritmo TS utiliza procedimentos de memória e processos de aprendizagem.

A memória utilizada pelo TS pode ser explícita ou implícita, baseada em atributos, embora ambas não sejam excludentes. Na memória do tipo explícita conservam-se soluções completas que consistem numa elite de soluções visitadas durante a busca, ou vizinhanças altamente atrativas porém inexploradas para tais soluções. Estrategicamente essas soluções especiais são introduzidas para ampliar $N^*(x)$ e assim apresentar opções úteis que não se encontram em $N(x)$.

A memória do TS pode ser feita de modo mais refinado através do uso de atributos, que guardam informações sobre características das soluções que se alteram ao efetuar-se um movimento de uma solução para outra.

Na situação de grafos ou redes, os atributos podem constituir-se de nós ou arcos que se suprimem ou se substituem pelos movimentos executados. Em problemas mais abstratos os atributos podem corresponder a valores das variáveis ou funções. Às vezes, os atributos podem combinar-se estrategicamente, de modo a criar outros atributos, mediante procedimentos por segmentações relacionadas com a Inteligência Artificial (AI).

Apresentam-se a seguir, de modo mais detalhado, as características do TS:

5.1 Memória de curto prazo

A memória de curto prazo é o tipo de memória mais comumente utilizado pelos algoritmos TS e leva em conta os atributos da solução que foram alterados no passado recente sendo chamada de memória baseada em ocorrências recentes.

Os atributos selecionados que apresentam as soluções visitadas recentemente são designados como tabu-ativos e as soluções que contém elementos tabu-ativos ou combinações particulares desses atributos são as que se convertem em tabu. Isto evita que algumas soluções do passado recente pertençam a $N^*(x)$ e portanto sejam revisitadas. Também se evita que outras soluções que compartilhem atributos tabu-ativos sejam revisitadas. O uso de avaliações tabus, que assumem penalidades grandes para conjuntos apropriados de atributos tabu-ativos, permite que o estado proibido varie dentro de uma gradação.

5.1.1 Manipulação da memória baseada em eventos recentes

A manipulação da memória é feita através da criação de uma ou mais listas de proibição nas quais são registrados os atributos tabu-ativos que, explícita ou implicitamente, identificam seu estado atual.



Denomina-se frequência tabu a duração que um atributo permanece tabu-ativo (medido em número de iterações). A frequência permite a criação de diferentes tipos de ajustes entre as estratégias de curto e longo prazo.

5.1.2 Critério de aspiração

Um outro elemento importante do TS é o critério ou nível de aspiração que consiste em permitir que um estado proibido seja ignorado se certas condições são cumpridas, dando flexibilidade ao algoritmo. É óbvio que qualquer solução que seja melhor do que qualquer das anteriormente encontradas, deve ser considerada como admissível, ainda que para alcançá-la devamos lançar mão de um movimento proibido. Critérios similares para a qualidade das soluções produzem critérios de aspiração para subconjuntos de soluções que pertençam a regiões comuns ou que compartilhem características específicas (tais como um valor funcional particular ou um nível de impossibilidade).

5.1.3 Estratégias para a lista de candidatos

As estratégias utilizadas para se obter a lista de candidatos é uma outra característica do TS. O caráter agressivo do TS é reforçado pela busca do melhor movimento disponível que pode ser determinado com uma quantidade apropriada de esforço. O significado de melhor não está limitado somente à avaliação da função objetivo. Este significado pode ser afetado pelas penalidades ou incentivos citados anteriormente, determinados pela história da busca. As estratégias para se obter a lista de candidatos são utilizadas para restringir o número de soluções a serem examinadas em uma dada iteração, para os casos em que $N^*(x)$ é muito grande ou quando a avaliação de seus elementos é muito custosa.

É muito importante para o TS uma seleção criteriosa dos elementos para o processo de busca, o que torna crítico contar com regras eficientes para a geração e avaliação de bons candidatos, ainda que nos casos onde não se utilizam explicitamente as estratégias para a lista de candidatos, as estruturas de memória que dêem atualizações eficientes das avaliações do movimento de uma iteração para outra, e que reduzam o esforço para encontrar melhores ou quase melhores movimentos, são parte integral das implementações TS. A atualização inteligente



pode reduzir bastante os tempos de resolução, e a inclusão de estratégias para a lista de candidatos para problemas grandes, pode aumentar significativamente os benefícios resultantes.

A memória de longo prazo dá ao algoritmo TS uma sofisticação maior e um desempenho melhor e possui três aspectos fundamentais:

Ela pode estar baseada em frequência, pode ter uma estratégia de intensificação e uma estratégia de diversificação.

A memória baseada em frequência consiste em armazenar o número de vezes em que um atributo foi escolhido para participar das configurações durante o processo TS. Pode estar baseada em frequências de transição que é o armazenamento do número de vezes que um atributo é retirado ou adicionado para formar novas configurações ou baseada em frequência de permanência que é o armazenamento do número de vezes em que um atributo permanece nas novas configurações ou em todas as configurações geradas durante o processo. A partir dessa informação pode-se penalizar ou incentivar certas configurações.

A estratégia de intensificação consiste em alterar os critérios de seleção de novas configurações para privilegiar certas soluções, podendo ser implementada:

- eliminando vizinhos ou incorporando novos vizinhos para a configuração corrente.
- modificando a caracterização da configuração vizinha.
- retornando às regiões atrativas para realizar uma busca mais acurada.
- realizando intensificação por decomposição.

A estratégia de diversificação tem como objetivo levar a busca para regiões novas e atrativas, mudando a definição de vizinhança ou de configurações candidatas.

A oscilação estratégica procura evitar que a busca se dê de modo monótono e consiste em fazer interações entre intensificação e diversificação a médio ou longo prazo, quando se atinge um nível crítico no qual o método TS normalmente pararia. Assim, ao invés de parar, mudam-se as regras de escolha de novas configurações candidatas para sair fora do nível crítico e, uma vez ultrapassado, volta-se novamente às regras anteriores de escolha dos candidatos. A repetição desta estratégia produz um movimento oscilatório em torno do nível crítico.

Outra estratégia que pode ser utilizada pelo TS é a construção de trajetórias que unam configurações de elite, chamada de *path relinking*, ou ligação de



caminhos. A idéia básica dessa estratégia é obter configurações constituídas por atributos de alta qualidade pertencente a configurações diversas também de alta qualidade.

5.2 Memória de longo prazo

Dependendo da aplicação, a memória de curto prazo do TS já é suficiente para produzir soluções de alta qualidade, porém pode ser incluída uma memória de longo prazo e suas estratégias associadas, de modo a tornar o TS mais eficiente.

Nas memórias de longo prazo são fundamentais as implementações baseadas em frequência, que introduzem penalidades ou incentivos em função do tempo durante o qual o atributo permanece nas soluções visitadas durante a busca, permitindo diferenciar as regiões.

As frequências de transição mantêm um registro da frequência com que se alteram os atributos, mesmo que a frequência de permanência mantenha os registros das durações relativas dos atributos das soluções geradas.

Suas melhoras começam a se manifestar em um prazo relativamente curto e permite chegar à solução final um pouco mais rápido comparativamente a implementações que utilizam apenas memória de curto prazo.

5.2.1 Estratégias de intensificação

Constituem um componente muito importante na implementação das memórias de longo prazo juntamente com as estratégias de diversificação.

Elas estão baseadas na modificação das regras de escolha de modo a favorecer combinações de movimentos e soluções com características que historicamente tenham sido boas. Podem provocar um retorno até regiões atrativas para se efetuar uma busca mais detalhada ou intensificada.

Duas variantes de intensificação tem tido muito êxito: uma consiste na introdução de uma medida da diversificação para garantir que as soluções armazenadas difiram umas das outras de um grau desejado e, antes de retornar ao processo a partir da melhor solução obtida, se limpar toda a memória de curto prazo; outra consiste na manutenção de uma lista seqüencial, de tamanho limitado, que

aceite ao final uma nova solução somente se ela for melhor que qualquer outra visitada previamente.

Normalmente é escolhido o último elemento da lista atual como base para se reiniciar a busca, sendo ele então suprimido. A memória de curto prazo do TS, que acompanha esta solução, também é armazenada e o primeiro movimento inibe assim o movimento previamente tomado desta solução, com o que um novo caminho de solução é iniciado. Algumas vezes este enfoque de recuperar soluções de elite é chamado de *backtracking* (busca reversa) nos métodos de busca em árvores e simplesmente consiste em criar uma lista limitada de prioridade.

Esta segunda variante está relacionada com a estratégia de reiniciar a busca desde vizinhanças não visitadas, previamente geradas. Tal estratégia mantém um registro da qualidade destas vizinhanças para selecionar um conjunto de elite e limita a atenção para tipos específicos de soluções, como são as vizinhanças de ótimos locais ou vizinhanças de soluções visitadas em passos imediatamente anteriores, antes de se alcançar tais ótimos locais. Este tipo de estratégia de “vizinhança não visitada” tem sido pouco examinado. As duas variantes acima têm produzido soluções de alta qualidade.

Outra estratégia de intensificação é a “intensificação por decomposição” na qual se impõem restrições à parte do problema ou à estrutura da solução para gerar uma forma de decomposição que permita um enfoque mais concentrado em outras partes da estrutura.

5.2.2 Estratégias de diversificação

As estratégias de diversificação são projetadas para conduzir a busca para novas regiões e, em geral, procuram modificar as regras de seleção para levar a busca para soluções cujos atributos não tenham sido usado freqüentemente. Alternativamente, pode-se introduzir estes atributos ao reiniciar parcial ou totalmente o processo de busca das soluções.

Os ótimos locais alcançados e utilizados como base para se executar uma seqüência de passos com o objetivo de diversificação pode diferir dos verdadeiros ótimos locais já que as regras de seleção do TS podem excluir alguns movimentos otimizadores. O êxito deste enfoque tem o mérito de incorporar uma variante de TS que caminha sempre para um verdadeiro ótimo local, uma vez que um movimento



otimizador se converte em uma seleção aceitável (baseado no critério de aspiração que é ativado somente depois de executar um movimento otimizador). Neste enfoque, ainda que existam movimentos que adicionalmente melhorem a seleção, o critério de aspiração permitirá que um deles seja selecionado mediante uma regra de avaliação tabu, que penalizará as opções baseando-se no estado tabu.

Uma vez que um verdadeiro ótimo local é alcançado, o critério de aspiração especial se interrompe, até que um novo movimento otimizador seja selecionado utilizando as regras de busca tabu padrão.

5.3 Oscilação estratégica

A oscilação estratégica proporciona um meio de fazer uma interação mais efetiva entre a intensificação e a diversificação seja a médio ou longo prazo. Ela atua orientando os movimentos em relação a um certo nível crítico, identificando por uma etapa de construção ou por um intervalo dado de valores para a função.

Tal nível crítico representa um ponto onde o algoritmo se encerraria normalmente. Ao invés de encerrar, ao alcançar este nível, as regras de seleção dos movimentos se modificam, para permitir que a região definida pelo nível crítico seja ultrapassada. O enfoque permite então chegar até uma certa profundidade previamente especificada mas até o nível crítico e então regressar. A busca continua agora até alcançar novamente o nível crítico e ultrapassá-lo, só que desta vez se dará em direção oposta a que tenha sido feita anteriormente e o processo se dirige a um novo ponto de retorno.

O processo de aproximar-se e ultrapassar uma e outra vez o nível crítico em direções diferentes cria um comportamento oscilatório, sendo daí o seu nome. Pode-se estabelecer um controle sobre este comportamento gerando avaliações e regras modificadas de movimento, dependendo da região explorada e a direção da busca. A possibilidade de percorrer novamente uma trajetória é evitada mediante os mecanismos usuais da busca tabu.

5.4 Reencadeamento de trajetórias (*path relinking*)

Este procedimento proporciona uma integração das estratégias de intensificação e diversificação, baseando-se na exploração de trajetórias que



conectam soluções de elite para gerar novas soluções, iniciando-se numa destas soluções chamada de solução inicial e gerando uma trajetória no espaço de vizinhanças que conduzem a outras soluções, chamadas soluções guias. Isto é feito selecionando-se movimentos que introduzem atributos contidos nas soluções guias.



CAPÍTULO 6

6. ALGORITMO TABU SEARCH PARA O PTCF

Esse procedimento heurístico TS foi desenvolvido para resolver o problema PTCF balanceado, ou seja, com a soma das demandas igual a soma das ofertas e, embora descrito para esse problema, pode ser aplicado a qualquer problema de custo mínimo de fluxo de rede ou problema de fluxo a custo mínimo (PFCM) com custos fixos, em virtude do procedimento heurístico empregado utilizar o método simplex num grafo como método de busca local. A implementação do TS aqui apresentada, se baseia na publicação de SUN et al. (1998:441-456).

O problema PTCF, como já descrito anteriormente, é formalizado como:

$$\min \sum_{i=1}^m \sum_{j=1}^n g_{ij}(x_{ij}) \quad (6.1)$$

$$sa \quad \sum_{i=1}^m x_{ij} = d_j \quad j=1,2,\dots,n \text{ (demandas)} \quad (6.1a)$$

$$\sum_{j=1}^n x_{ij} = s_i \quad i=1,2,\dots,m \text{ (ofertas)} \quad (6.1b)$$

$$x_{ij} \geq 0 \quad i=1,2,\dots,m \quad e \quad j=1,2,\dots,n \quad (6.1c)$$

sendo

$$g_{ij}(x_{ij}) = 0 \quad se \quad x_{ij} = 0 \quad (6.1d)$$

ou

$$g_{ij}(x_{ij}) = c_{ij}x_{ij} + f_{ij} \quad se \quad x_{ij} > 0 \quad 6.1e)$$

Equação 6.1 – PTCF – Problema de Transporte com Carga Fixa – SUN et al. (1998)

6.1 Conceitos de PTCF e Grafos

O PTCF consiste numa rede de transportes com nós de origens e nós de destino, que devem ser conectados com arcos de modo unidirecional. Podemos considerar este conjunto de nós de origens e de destinos como um Grafo, que consiste num conjunto de nós ou vértices. Tem-se um Grafo Bipartido quando existe um número N finito de nós (vértices ou pontos) e um conjunto A de arcos ligando pares de nós. No Grafo Bipartido tem-se uma capacidade, $s_i > 0$, associada a cada

nó i de origem e uma demanda solicitada, $d_j > 0$, associada a cada nó j de destino.

Nos problemas que envolvem decisão quanto à conexão entre origens e destinos, segundo GOLDBARG e LUNA (2000), existem três tipos de estruturas extremamente importantes:

Caminhos, nos quais se tem o foco na trajetória; árvores, nas quais o foco está na continuidade da conexão; e emparelhamento, no qual foco está na conexão entre pequenos grupamentos.

Dois exemplos de problemas que ocorrem num grafo;

- Problemas de fluxo de custo mínimo (PFCM), que aparecem em redes logísticas, onde materiais ou pessoas são movimentados entre vários pontos.

- Problemas do caminho mais curto (PCMC), nos quais, para dois vértices do Grafo (N,A) , um de partida e um de chegada, se busca encontrar o caminho mais curto.

Alguns conceitos de problemas de fluxo em rede sobre um grafo são destacados abaixo, segundo BAZARAA, JARVIS e SHERALI (1990):

- uma árvore (*tree*) é constituída por um grafo conectado e sem ciclos; uma árvore geradora (*spanning tree*) é uma árvore que contém todos os nós do grafo enquanto um a árvore geradora mínima (*rooted spanning tree*) é uma árvore cuja soma das arestas é mínima em G , tendo no caso do PTCF ou do PT, exatamente $m+n-1$ arcos.

- um caminho ou percurso do nó i para o nó j (*path*) é uma seqüência de arcos nos quais o nó inicial de cada arco é o mesmo que o nó final do arco precedente, sendo os nós, na seqüência, todos distintos.

- uma cadeia ou corrente (*chain*) é uma estrutura similar a um caminho exceto quanto ao fato que nem todos os arcos são direcionados para o nó final.

- um circuito (*circuit*) é um caminho de um nó inicial até um nó final, associado ao arco que interliga o nó final com o nó inicial, fechando o caminho assim constituído.

- um ciclo (*cycle*) é um caminho fechado.

No PTCF, tem-se que uma rota ou conexão da origem i para o destino j , é chamada de arco na rede de transportes e representado por (i, j) . Cada localização de origem ou destino é representada por um nó. Existem m nós de origem e n nós de destino; sendo s_i a capacidade disponível do nó i ; d_j a demanda do nó; c_{ij} são os

custos por unidade transportada e f_{ij} são os custos fixos associados, respectivamente, ao estabelecimento da rota ou arco (i,j) .

Assume-se que $f_{ij} \geq 0$, para cada $i=1,2,\dots,m$, $j=1,2,\dots,n$, e que a variável de decisão $x_{ij} \geq 0$ representa o número de unidades transportadas (também chamada de fluxo) na rota correspondente ao arco (i,j) . Na implementação efetuada não foram considerados limites superiores para os valores de x_{ij} , mas poderiam ser implementado no algoritmo TS.

No caso em análise a rede não é totalmente conectada em virtude de não existirem arcos ou rotas estabelecidas entre todas as origens e todos os destinos, e considera-se que as somatórias das relações (6.1a) e (6.1b) são para os valores x_{ij} para os quais existem rotas entre a origem i e o destino j .

Têm-se $nb=m+n-1$ arcos de conexão na rede de transporte, constituindo uma base ou árvore geradora mínima para o problema, contra um total de $N=m \times n$, correspondente ao caso da rede ser totalmente conectada.

O problema PTCF é formulado e resolvido, freqüentemente, como um problema de programação de fluxo de rede inteiro misto mas, em princípio, poderia ser utilizado qualquer método de resolução para programação inteira mista geral para resolvê-lo, porém, segundo STEINBERG (1970), estes métodos são geralmente ineficientes e computacionalmente pesados porque não tiram vantagem da estrutura especial da rede de transporte do problema.

6.2 Algoritmo TS para o PTCF

No procedimento TS implementado, utilizaram-se três componentes básicos: **processo de memória de curto prazo**, **processo de memória intermediária** e **processo de memória de longo prazo**, que são acionados em função do número de iterações sem que ocorra alteração no valor da solução corrente, conforme mostrado no fluxograma da fig. 6.1.

O processo de **memória de curto prazo** é baseado numa **lista tabu** (conjunto de condições proibidas) e num **critério de aspiração**. Com base nas memórias baseadas de ocorrências recentes e em freqüência que definem a busca tabu, mantém-se como proibido de ser executado um subconjunto de movimentos potenciais, evitando-se assim a repetição ou reversão para soluções visitadas previamente.



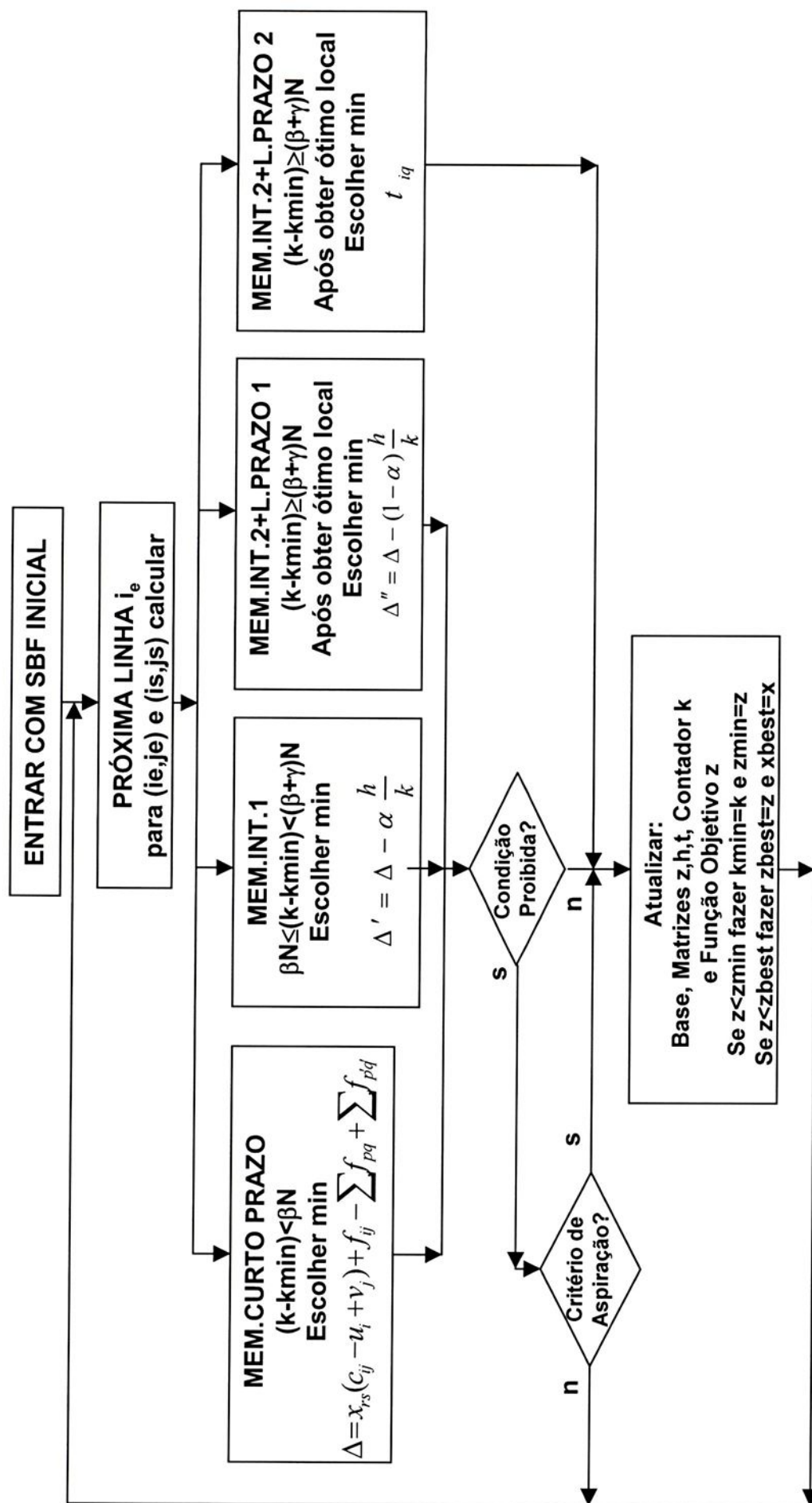


Figura 6.1 – Fluxograma Simplificado do Algoritmo TS

O processo de **memória intermediária** é implementado pela restrição do espaço de busca dentro de um conjunto de soluções potencialmente prósperas para a **intensificação** da busca.

O processo de **memória de longo prazo** é solicitado periodicamente para conduzir a busca para novas regiões que ainda não foram exploradas com a finalidade de **diversificar** a busca.

O procedimento de TS implementado explora a estrutura de rede especial dos problemas PTCF, e se baseia no método simplex para problemas de transporte padrão como algoritmo de busca local, para levar a busca de uma solução básica factível para outra.

6.2.1 Obtenção de uma SBF inicial

No algoritmo TS desenvolvido e implementado computacionalmente, optou-se pela utilização de uma SBF inicial obtida externamente, ou seja, utiliza-se uma configuração básica factível inicial, que seja um ótimo local do PTCF, mas não obrigatoriamente, obtida por um algoritmo de programação linear especializado, e implementado computacionalmente, para o problema de transporte PT obtido por uma adaptação da heurística de BALINSKI (1961) em conjunto com a Regra do Noroeste ou do Custo Mínimo para obter uma solução de partida.

A solução ótima para o problema relaxado fornece uma solução factível inicial para o problema PTCF com um valor da função objetivo igual a:

$$z^* = \sum_{i=1}^m \sum_{j=1}^n (c_{ij} x_{ij} + f_{ij} y_{ij}) \quad (6.2)$$

$$\text{sendo } y_{ij} = 0 \quad \text{se } x_{ij} = 0 \quad (6.2a)$$

e

$$y_{ij} = 1 \quad \text{se } x_{ij} > 0 \quad (6.2b)$$

Equação 6.2 - Algoritmo TS - Função objetivo para a SBF inicial para o TS

BALINSKI (1961) mostrou que o **valor ótimo** da função objetivo para o **problema relaxado** fornece um **limite inferior** para o valor da função objetivo do problema PTCF **original**, e z^* definido na igualdade (6.2) fornece um **limite superior**.

6.2.2 Funcionamento do Algoritmo TS para o PTCF

A solução gerada pelo processo anterior constitui uma solução factível básica inicial para inicialização do algoritmo TS, sendo essa SBF um ótimo global do problema PT relaxado, correspondendo a um ótimo local para o PTCF, a partir do qual se dá início ao procedimento TS específico.

A linha básica utilizada na implementação do TS específico para o PTCF é a do método simplex para o problema de fluxo em rede, levando-se em conta os custos fixos. BAZARAA, JARVIS E SHERALI (1990:434-440) descrevem o problema de transporte padrão, utilizando fluxo em rede, sem levar em conta os custos fixos, e servem de base para as rotinas aqui implementadas, incorporando-se o custo fixo, de modo adequado, ao processo de decisão dos movimentos e ao funcionamento do TS.

Essa abordagem de Busca Tabu para o PTCF utiliza, no início do processo, um procedimento chamado de **memória de curto prazo** onde a busca é bastante semelhante ao proposto por BAZARAA, JARVIS E SHERALI (1990) no método simplex específico para o problema de transporte padrão, sem custos fixos.

As rotinas de armazenamento de informações são feitas de modo implícito, e permitem alterar o curso da busca, em função das mudanças ocorridas mais recentemente ou mais freqüentemente na base, tendo sido desenvolvidas duas estratégias para cada um dos processos decorrentes dessas informações: **processo de memória intermediária 1 e 2** e **processo de memória de longo prazo 1 e 2**.

O algoritmo consiste em se fazer, inicialmente, leitura dos dados de entrada para o PTCF. A partir de um arquivo pré-gravado, que utiliza o formato padrão FCTP, obtido de GOTTLIEB (2001), onde se especificam o número de fornecedores, o número de destinos, a capacidade dos fornecedores, a demanda dos destinos, a matriz de custos unitários de transportes e finalmente a matriz de custos fixos relativos à abertura das fontes de fornecimento.

A seguir é efetuada a leitura do arquivo gerado anteriormente pela heurística e PPL baseados em BALINSKI, que fornece a SBF inicial, no formato de uma matriz $m \times n$ com $m+n-1$ valores das quantidades a transportar, $x_{ij} \geq 0$, correspondentes aos valores atribuídos. Adotou-se neste trabalho a representação das variáveis não básicas, que tem valor $x_{ij} = 0$, com a atribuição $x_{ij} = -1$, para diferenciá-las das

variáveis básicas que apresentam valor nulo, o que ocorre nas soluções degeneradas.

Após a leitura desses arquivos, adotou-se um armazenamento compacto da base, com índices de 1 a $nb=m+n-1$, armazenando em vetores, com nb valores, para a linha (li), para a coluna (co), para o valor atribuído básico (xb), para o custo unitário (cb) e para o custo fixo (fb). Assim, para cada índice k , variando de 1 até nb , tem-se um conjunto (li,co,xb,cb,fb) que define um elemento básico, e considera-se qualquer (i,j) com $x_{ij} = -1$ na matriz (x_{ij}) tipo $m \times n$, como um elemento não pertencente à base.

Este armazenamento compacto facilita o pivoteamento e a atualização da base, cujo processo mais complexo é o da determinação de um conjunto de vetores básicos que sozinhos não formem um ciclo e, em conjunto com o arco não básico candidato a entrar na árvore, forme um ciclo. Este procedimento é adotado para cada vetor não básico candidato a entrar na base, na linha corrente. O armazenamento compacto também facilita o cálculo, de modo rápido, do valor da função objetivo corrente, após cada mudança de base.

Na busca de uma solução de melhor qualidade adota-se o procedimento de se colocar as variáveis não básicas em função das variáveis básicas para o PT e adaptado para o PTCF, chamado do método do ciclo ou do percurso de avaliação.

BRONSON (1985) descreve o percurso de avaliação como uma seqüência de arcos ou células tais que:

- todo par de arcos consecutivos está ou sobre a mesma linha ou sobre a mesma coluna.
- não existem três arcos consecutivos sobre a mesma linha ou coluna.
- o primeiro e o último arco da seqüência estão sobre a mesma linha ou coluna.
- nenhum arco figura mais de uma vez, na seqüência.

6.2.3 Cálculo da variação da função objetivo

Para se decidir sobre as mudanças de uma SBF para outra, calculam-se inicialmente para a SBF inicial e após cada mudança da base, os valores das variáveis duais u_i e v_j associadas a cada nó. Estes valores são utilizados na determinação da variação líquida da função objetivo.



Para cada variável não básica (i_e, j_e) , da linha corrente i_e , candidata a entrar na base, e para a sua correspondente variável básica (i_s, j_s) , determinada em cada caso, a partir da construção de um caminho aberto, sem formar ciclo, de variáveis básicas, partindo da linha i_e para a coluna j_e , calcula-se a variação da função. O arco (i_s, j_s) escolhido provisoriamente para sair da base é aquele que apresenta o menor valor entre os x_{pq} , pertencentes ao caminho determinado e de modo a manter a factibilidade do problema após a troca da base.

Inicialmente, calcula-se para cada elemento não básico (i_e, j_e) , candidato a entrar na base, da primeira linha da matriz x das soluções, o valor da variação da função objetivo correspondente a entrada do elemento básico (i_s, j_s) , determinado no ciclo. A linha corrente é incrementada de um em um de modo cíclico, voltando a linha 1 ao se ultrapassar a última linha da matriz, durante todo o processo memória de curto prazo.

Assim, para a linha i_e corrente, para cada elemento não básico (i_e, j_e) , tem-se um candidato a entrar na base e, a partir dele, determina-se a variável (i_s, j_s) , determinado de modo único, a sair da base. Determina-se o ciclo fechado formado pela variável candidata a entrar (i_e, j_e) e pelos elementos básicos, partindo da origem i_e até se chegar ao destino j_e . Calcula-se a variação da função objetivo levando-se em conta o valor das variáveis de decisão numa base provisória e calcula-se, para todos os elementos não básicos da linha a variação líquida da função objetivo, como se a troca fosse feita.

No processo memória de curto prazo a decisão de mudança ocorre diretamente a partir do cálculo da variação líquida da função objetivo:

$$\Delta_{ij} = 0 \quad \text{se} \quad x_{rs} = 0 \quad (6.3a)$$

ou

$$\Delta_{ij} = x_{rs} \cdot (C_{ij} - u_i + v_j) + f_{ij} - \sum f_{pq} + \sum f_{p'q'} \quad \text{se} \quad x_{rs} > 0 \quad (6.3b)$$

Equação 6.3 - Algoritmo TS - Variação da função objetivo

sendo:

u_i é a variável dual associada ao nó i , para a SBF corrente.

v_j é a variável dual associada ao nó j , para a SBF corrente. .

(p, q) são os arcos que estão na base tais que x_{pq} eram diferentes de zero e assumem o valor 0 após a troca.

(p', q') são os arcos que estão na base tais que $x_{p'q'}$ saiam de zero e vão para um valor maior que 0 após a troca

c_{ij} é o custo unitário de transporte do nó i para o nó j

f_{ij} é o custo fixo de abertura da rota (i, j)

x_{rs} é o menor valor entre os valores do ciclo que sofrerão redução do valor x_{rs} até 0 ao sair da base.

Se o valor $\Delta_{ij} \leq 0$, tem-se uma melhoria no valor da função objetivo corrente z_c , e completa-se a troca da variável (i_e, j_e) entrando na base pela variável (i_s, j_s) saindo da base, e atualizam-se os valores da matriz (x_{ij}) das quantidades transportadas, independentemente de outras condições, dentro do processo memória de curto prazo.

Nos outros processos o critério de seleção é alterado em função de se buscar a intensificação da busca ou de se buscar a diversificação para regiões ainda não muito exploradas pela busca porém, caso ocorra efetivamente a troca, o cálculo da alteração do valor da função objetivo é feito pela relação (6.3)

O critério para definir (i_e, j_e) e (i_s, j_s) é alterado de modo dinâmico guiado pelas informações sobre o histórico da busca armazenado pelo TS e pelos parâmetros de duração dos processos correspondentes as várias memórias implementadas, e utilizando-se diferentes critérios para tornar certas configurações mais atrativas em função de terem permanecido mais vezes na base no caso do processo de intensificação ou menos atrativas no caso do processo de diversificação.

Na implementação computacional define-se:

z_c = valor da função objetivo da solução corrente.

k = número da iteração corrente.

Se τ é a base ou árvore geradora da rede de transportes e se $(i_s, j_s) \in \tau$, tem-se que o arco (i_s, j_s) está atualmente na base ou seja, ele é um **arco básico**.

6.2.4 Movimento e memórias

Um movimento é definido como uma transição de uma solução básica factível para outra, ou seja, a mudança de uma base para outra, sendo equivalente a uma iteração ou uma pivotagem no método simplex sobre um grafo.

Para implementar os processos de memória utilizam-se três matrizes: **z**, **t** e **h**.

A matriz **z** representa o valor da função objetivo a cada iteração

A matriz **t** representa a memória baseada em ocorrências recentes.

A matriz **h** representa a memória baseada em frequência.

Os elementos (i,j) dessas três matrizes são definidos como segue:

z_{ij} = valor da função objetivo imediatamente antes de (i,j) sair da árvore geradora caso $(i,j) \notin \tau$, ou imediatamente antes de (i,j) ter entrado para a árvore geradora caso $(i,j) \in \tau$.

t_{ij} = número da iteração imediatamente antes de (i,j) sair da árvore geradora se $(i,j) \notin \tau$, ou imediatamente antes de (i,j) ter entrado para a árvore geradora se $(i,j) \in \tau$.

h_{ij} = frequência com que $(i,j) \in \tau$, isto é, o número de iterações para as quais (i,j) permanece básico.

Se (i,j) nunca entrou ou nunca saiu da árvore geradora desde que o processo de busca iniciou, têm-se os seguintes valores iniciais:

z_{ij} = valor da função objetivo da primeira solução ótima local

$t_{ij} = -1e9$, ou seja, é fixado para um valor suficientemente negativo.

$h_{ij} = 0$, ou seja, é fixado para um valor igual 0 uma vez que nenhuma configuração foi montada.

Para cada (i,j) que seja básico ou não básico, são mantidos três valores associados z_{ij} , t_{ij} e h_{ij} .

Se, numa próxima iteração, o movimento for incluir $(i_e, j_e) \notin \tau$ e retirar $(i_s, j_s) \in \tau$ então o movimento fica caracterizado pela **8-upla seguinte**:

$$((i_e, j_e), (i_s, j_s), z_{ij}, z_{rs}, t_{ij}, t_{rs}, h_{ij}, h_{rs}) \quad (6.4)$$

Equação 6.4 - Algoritmo TS - Movimento - 8-upla

Os valores z_{ij} e t_{ij} são atualizados sempre que (i,j) entra ou sai da árvore geradora, enquanto que os valores h_{ij} são atualizados somente quando (i,j) sai da árvore geradora.

6.2.5 Soluções vizinhas e avaliação de movimentos.

Dada uma solução básica factível corrente, qualquer $(i,j) \notin \tau$ pode ser introduzido na árvore geradora para passar dessa solução básica factível corrente para uma solução básica factível adjacente, sendo cada uma dessas transições um candidato para o próximo movimento.

A **vizinhança** de uma dada solução básica factível é composta por todas as soluções básicas factíveis que podem ser atingidas fazendo um único movimento, ou seja, uma única troca constituída pela substituição de $(i_s, j_s) \in \tau$ por $(i_e, j_e) \notin \tau$.

Segundo BAZARAA, JARVIS E SHERALI (1990:440) para o problema de transporte padrão (sem custo fixo) baseado em fluxo em rede, para se efetuar uma troca, primeiro deve ser respondida a pergunta: qual arco não básico entrará na base? Deverá entrar na base o arco não básico que tem o menor custo reduzido positivo calculado por $\Delta = u_i + v_j - c_{ij}$, aqui transformado para se trabalhar com valor negativo como $\Delta = c_{ij} - u_i - v_j$.

No algoritmo TS desenvolvido o processo funciona de modo semelhante, ou seja, é escolhido um arco $(i,j) \notin \tau$ não básico para entrar na árvore na linha corrente que é incrementada uma a uma de modo cíclico segundo o algoritmo iterativo, e testa-se após obter um provável candidato (i_s, j_s) a sair, se (i_e, j_e) realmente entra ou não na base, na medida em que apresenta um menor valor negativo, calculado no algoritmo original de SUN et al (1998) como $\delta(c_{ij} - \pi_i + \pi_j)$, aqui substituídas por $x_{rs}(c_{ij} - u_i + v_j)$ na relação (6.3) correspondendo à parcela dos custos unitários e acrescentando-se a parcela do custo fixo correspondente a variável que entra na base com quantidade maior que zero e descontando-se os custos fixos das variáveis que já estão na base com valor positivo e que se tornam iguais a zero após a iteração.

Se o arco não básico $(i,j) \notin \tau$ é escolhido para ser introduzido na árvore geradora no próximo movimento, determina-se um caminho alternativo (ciclo) $\rho \subset \tau$ do nó i para o nó j e, de modo semelhante aos algoritmos simplex para fluxos em rede. O arco saindo pertencente a este ciclo $(i_s, j_s) \in \rho$ é determinado pelo teste abaixo de modo a manter a factibilidade, sendo o arco saindo (i_s, j_s) aquele para o qual o valor x_{rs} foi encontrado.

$$x_{rs} = \min \{ x_{pq} \mid (q,p) \in \rho \text{ e } x_{pq} \text{ sofre redução} \} \quad (6.5)$$

Equação 6.5 - Algoritmo TS - Determinação do arco saindo da base

Steinberg (1977) sugere que quando $x_{rs} = 0$, isto é, degenerado, se escolha $(i_s, j_s) \in \rho$ como o arco saindo da árvore geradora aquele que tem o maior custo fixo entre todos os arcos elegíveis para sair, o que foi aqui implementado.

No problema de transporte padrão, conforme BAZARAA, JARVIS E SHERALI (1990;491), o valor Δ tem um significado diferente do Δ_{ij} do TS ora implementado. Ele é definido como o menor valor entre os x_{ij} que sofrerão redução de positivo para zero entre os arcos básicos pertencentes ao ciclo fechado formado pelo arco que vai entrar na base e os arcos da árvore ou base.

A partir daí encontra-se uma nova solução atualizando-se o valor pela expressão $x_{ij}(\text{novo}) = x_{ij}(\text{anterior}) - \Delta$, atualizando-se os demais valores da base.

No algoritmo TS, ao se executar o movimento tem-se o novo valor de x_{ij} :

$$x_{ij}' = x_{rs} \quad (6.6)$$

Equação 6.6 - Algoritmo TS - Movimento - novo valor de x_{ij}

e a mudança líquida da função objetivo será:

$$\Delta_{ij} = 0 \quad \text{se} \quad x_{rs} = 0 \quad (6.7a)$$

ou

$$\Delta_{ij} = x_{rs}(c_{ij} - u_i + v_j) + f_{ij} - \sum_{(p,q) \in U1} f_{pq} + \sum_{(p',q') \in U2} f_{p'q'} \quad \text{se} \quad x_{rs} > 0 \quad (6.7b)$$

Equação 6.7 - Algoritmo TS - Movimento - Mudança líquida da função objetivo

sendo:

$$(p,q) \in U_1 = \{ (p,q) \mid (q,p) \in \rho \text{ e } x_{pq} = \delta \}$$

$$(p', q') \in U_2 = \{ (p', q') \mid (p', q') \in \rho \text{ e } x_{p'q'} = 0 \}$$

u_i = variável dual associada à linha i .

v_j = variável dual associada à coluna j .

As variáveis duais são definidas e calculadas levando-se em conta apenas os custos variáveis.

Os termos $\sum_{(p,q) \in U_1} f_{pq}$ e $\sum_{(p',q') \in U_2} f_{p'q'}$ são os ajustes dos custos fixos para a variável saindo da base e para as variáveis degeneradas que refletem a mudança atual na função objetivo.

Se (i,j) entra na árvore geradora, U_1 irá incluir todos arcos básicos no ciclo obtido ρ cujos fluxos irão decrescer de positivo para 0 e U_2 irá incluir todos aqueles que irão crescer de 0 para positivo, respectivamente.

Conseqüentemente, os custos fixos serão subtraídos do valor corrente da função objetivo para os arcos que estão em U_1 e somados para os arcos que estão em U_2 .

A mudança líquida no valor da função objetivo, Δ_{ij} , definida na igualdade (6.7), mede diretamente durante o processo memória de curto prazo a atratividade do movimento provocado por $(i,j) \notin \tau$ para entrar na árvore geradora.

Para a busca localizada, encontra-se o melhor movimento pela escolha de $(i,j) \notin \tau$ para entrar na base, tal que:

$$\Delta_{ij} = \min \{ \Delta_{pq} \mid (p,q) \notin \tau \} \quad (6.8)$$

Equação 6.8 - Algoritmo TS - Movimento - Determinação do melhor movimento

Se $\Delta_{ij} \geq 0$ para todo $(i,j) \notin \tau$, a solução corrente é um **ótimo local**.

Porém no TS, ao contrário do algoritmo simplex para o PPL padrão, não é obrigatório se ter Δ_{ij} negativo para se fazer um movimento: para sair de uma solução ótima local, é necessário fazer movimentos buscando soluções que sejam piores que a solução corrente, diversificando a busca, de modo a se ultrapassar o ótimo local.

6.2.6 Condições proibidas (tabu)

No algoritmo desenvolvido mantém-se de modo implícito duas tabelas de condições proibidas (tabu), e definem-se os parâmetros ξ_{in} e ξ_{out} , utilizados para representar suas amplitudes.

ξ_{in} = número de iterações que um arco não básico permanece não básico antes de ter sido alocado para entrar para a árvore geradora.

ξ_{out} = número de iterações que um arco básico permanece básico antes de ser alocado para ser retirado da árvore geradora.

Para a decisão do movimento de substituição de $(i_s, j_s) \in \tau$ por $(i, j) \notin \tau$, são definidas as **condições proibidas** através da matriz t como segue:

$$k - t_{ij} \leq \xi_{in} \quad \text{ou} \quad k - t_{rs} \leq \xi_{out} \quad (6.9)$$

Equação 6.9 - Algoritmo TS - Condições Tabu

Se a desigualdade (6.9) é satisfeita, o movimento é considerado proibido e, caso contrário, o movimento pode ser realizado. Porém, se o movimento for proibido, ele só poderá ser feito se satisfizer ao **critério de aspiração**, descrito abaixo.

Como existem usualmente muito mais arcos não básicos do que arcos básicos, ξ_{in} é usualmente estabelecido com valores maiores que ξ_{out} .

Esses valores ξ_{in} e ξ_{out} podem ser estabelecidos de modo fixo ou variável durante a busca, de modo especificado ou aleatório.

Na implementação efetuada estabeleceram-se esses valores via teclado com valores obtidos experimentalmente e, ao longo do processo de diversificação, eles são alterados de modo sistemático de modo decrescente para ξ_{out} e de modo crescente para ξ_{in} , dentro de limites mínimo e máximo respectivamente.

Numa segunda fase desse trabalho, implementar-se-ão os valores de modo variável, dentro de faixas de valores obtidas experimentalmente entre um limite inferior e um limite superior tanto para ξ_{in} quanto para ξ_{out} .

No procedimento implementado, a cada chamada do processo de memória de longo prazo, são alterados os valores de ξ_{in} e ξ_{out} .

6.2.8 Critério de aspiração

O critério de aspiração é a estratégia utilizada para se desconsiderar os estados proibidos (tabu) de movimentos quando se encontram soluções melhores que o valor corrente. No algoritmo implementado adotou-se como **critério de aspiração** para o movimento de substituição de um arco $(i_s, j_s) \in \tau$ por $(i, j) \notin \tau$, utilizando-se os dados armazenados na matriz z :

$$z + \Delta_{ij} < \min \{ z_{ij}, z_{rs} \} \quad (6.10)$$

Equação 6.10 - Algoritmo TS - Critério de Aspiração

sendo:que, $z + \Delta_{ij}$ é o valor da função objetivo após o movimento proposto ser executado.

Se o movimento é tabu mas atende ao critério anterior, os estados proibidos de um movimento são desconsiderados, devido a solução resultante ser melhor que as soluções resultantes antes de (i,j) ser removido da base e antes de (i_s,j_s) ter sido incluído na base.

6.2.9 Lista de Candidatos

Utilizou-se, de modo implícito, uma abordagem baseada numa lista de candidatos, para se evitar que o tempo computacional seja excessivo na determinação dos Δ_{ij} para todos os $(i,j) \notin \tau$.

A cada iteração, todos os movimentos potenciais envolvendo arcos não básicos originários de uma única origem (linha) são colocados numa lista de candidatos e armazenados num vetor linha. A cada iteração atualiza-se esta lista com os novos candidatos. No caso, se o arco entrando foi escolhido entre todos os arcos $(i,j) \notin \tau$ para uma dada linha i na iteração corrente, o arco entrando na próxima iteração é escolhido entre os arcos $(i+1,j) \notin \tau$.

Se $(i + 1) > m$, ou seja, se o valor i ultrapassa o número de linhas da matriz, reinicia-se o processo na linha $i = 1$, de modo a se fazer uma varredura cíclica dos arcos com origens comuns.

Quando todos os movimentos na lista de candidatos são proibidos e nenhum deles satisfaz o critério de aspiração, muda-se a estratégia de modo que para cada iteração, o melhor movimento passe a ser selecionado de uma lista de candidatos e seus estados proibidos, que atendam as especificações da desigualdade (6.9).

Se o movimento não é proibido ele é realizado e. Do contrário, verifica-se o critério de aspiração especificado na desigualdade (6.10) é satisfeito, realizando-se o movimento mesmo que proibido. Caso não atenda ao critério de aspiração o procedimento continua sem alterar a base

Chama-se **movimento admissível** um movimento que não é tabu ou que, sendo tabu e satisfaça o critério de aspiração. No caso do movimento não ser admissível, ele simplesmente não pode ser feito, e o procedimento busca o próximo melhor movimento na mesma lista de candidatos.

O processo é repetido até que seja encontrado um movimento admissível ou até que tenham sido verificados todos os candidatos da lista para a linha i corrente.

Se nenhum dos movimentos na lista de candidatos é admissível, não é feito nenhum movimento e a lista de candidatos é reconstruída com todos os arcos originados na próxima origem. Desta maneira, permite-se que o processo de busca siga um padrão guiado pelo melhor movimento entre todos os movimentos admissíveis.

6.2.10 Função intensificação, função diversificação e critério de parada

Para implementar o **processo de memória intermediária** são utilizadas duas estratégias, que intensificam a busca regionalmente.

Na primeira estratégia, o processo de memória intermediária é chamado periodicamente por mudanças temporárias na medida de atratividade de um movimento de Δ_{ij} definido na igualdade (6.7) que passa a assumir o valor Δ'_{ij} .

$$\Delta'_{ij} = \Delta_{ij} - \alpha \frac{h_{ij}}{k} \quad (6.11)$$

Equação 6.11 - Algoritmo TS - Movimento - Determinação do melhor movimento no processo de memória intermediária

A constante $\alpha > 0$ é calculada de modo a assegurar que o valor $\alpha \frac{h_{ij}}{k}$ seja significativo mas não dominante na expressão de Δ'_{ij} .

O valor de α é computado para a primeira solução ótima local, na qual todos os Δ_{ij} para as variáveis não básicas são positivos, através da relação:

$$\alpha = \frac{\sum \Delta_{ij}}{N - m - n + 1} \quad \text{com } (i, j) \notin \tau \quad (6.12)$$

Equação 6.12 - Algoritmo TS - Cálculo do parâmetro α para o primeiro ótimo local

O valor Δ'_{ij} , definido pela igualdade (6.11), passa a ser a nova medida da atratividade e um movimento potencial trazendo determinado $(i, j) \notin \tau$ para a árvore geradora, para um dado (i, j) que tem estado na árvore mais frequentemente fica, por esse novo critério, relativamente mais atrativo devido ao fator da proporção $\frac{h_{ij}}{k}$ ser maior.

Desse modo faz-se uma **intensificação** da busca em torno das soluções cujos arcos estão mais freqüentemente na árvore geradora. Ao se encontrar uma melhoria na solução por esta estratégia, imediatamente a busca retorna ao processo de memória de curto prazo, dando continuidade ao algoritmo. Se não se encontra uma melhor solução após um certo número de movimentos aplicando essa estratégia, chama-se uma segunda estratégia de modo a ser buscar uma solução ótima local próxima.

Outro procedimento chamado **processo de memória de longo prazo** procura levar a busca para uma nova região que ainda pode não ter sido explorada, de modo a **diversificar** a busca. São implementadas duas estratégias para se implementar o processo de memória de longo prazo.

Numa primeira estratégia, utiliza-se a memória baseada em freqüência de modo similar a utilizada pelo processo de memória intermediária, na qual a medida da atratividade é mudada temporariamente de Δ_{ij}' definida pela igualdade (6.7) para Δ_{ij}'' :

$$\Delta_{ij}'' = \Delta_{ij}' - \alpha \left(1 - \frac{h_{ij}}{k}\right) \quad (6.13)$$

Equação 6.13- Algoritmo TS - Movimento - Determinação do melhor movimento no processo de memória de longo prazo 1

O valor Δ_{ij}'' definido na igualdade (6.13) passa a ser a nova medida da atratividade, de modo que um movimento potencial $(i,j) \notin \tau$ trazendo (i,j) para a base para um arco (i,j) que mais freqüentemente esteve na base dificilmente se torna relativamente mais atrativa, porque o valor $(1 - \frac{h_{ij}}{k})$ será um valor negativo, fazendo com que $-\alpha(1 - \frac{h_{ij}}{k})$ se torne positivo, de modo que Δ_{ij}'' será sempre maior do que Δ_{ij}' . Ao contrário, arcos que estão na árvore geradora menos freqüentemente, serão mais solicitados a entrar na árvore geradora.

Cada vez que se chama esta estratégia fazem-se m movimentos, um para cada origem, a partir da linha corrente.

Numa segunda estratégia, um $(i,j) \notin \tau$ que saiu da árvore geradora por longo período para cada origem i é introduzido na árvore geradora.

Para a origem i , (i,j) é selecionado a entrar na base se:

$$t_{ij} = \min \{t_{iq} \mid (i,q) \notin \tau, \quad 1 \leq q \leq n\} \quad (6.14)$$

Equação 6.14 - Algoritmo TS - Movimento - Determinação do melhor movimento no processo de longo prazo 2

Se na linha i existe mais de um arco que satisfaça a igualdade (6.14), seleciona-se o primeiro a ser encontrado. Deste modo, cada vez que este procedimento memória de longo prazo 2 é acionado, são trazidos m arcos para a árvore geradora, um para cada origem i .

Após executar o processo de memória de longo prazo, a busca retorna ao processo de memória de curto prazo.

Para que o processo de busca seja finito foi introduzido dentro do processo memória intermediária 2 um **critério de parada**:

Seja $\beta > 0$ o coeficiente tal que foi solicitado o processo de memória intermediária e se não for encontrada uma solução melhor após $\beta.N$ movimentos no processo de memória de curto prazo.

Seja $\gamma > 0$ o coeficiente tal que a primeira estratégia de memória intermediária executou $\gamma.N$ movimentos cada vez que foi solicitada e se não pode ser localizada uma solução melhor.

O procedimento executa as duas estratégias do processo memória de longo prazo que são solicitadas alternadamente, comandadas pela estratégia do processo de memória intermediária 2.

Um contador de iterações determina o número de vezes que o processo memória de longo prazo é acionado e, após o processo de longo prazo ser solicitado um número $L=l_{\max}$ de vezes, finaliza-se a busca.

Pela variação dos valores de β , γ e L , pode-se controlar o tamanho e o detalhamento da busca. Para valores pequenos pode-se concluir a busca rapidamente enquanto para valores maiores pode-se ter uma busca mais demorada e obtendo-se portanto melhores soluções. Através do ajuste adequado desses parâmetros pode-se ajustar o compromisso entre a qualidade da solução final e o tempo computacional.

Os valores ξ_{in} , ξ_{out} , L , β e γ são chamados de parâmetros do procedimento heurístico TS e, na implementação efetuada, são introduzidos via teclado na fase

inicial do algoritmo com valores escolhidos entre os valores determinados previamente de modo experimental.

6.2.11 Algoritmo TS para o PTCF

A seguir é descrito o procedimento TS para o PTCF, passo a passo, conforme descrito por SUN et al (1998) acompanhado de uma breve explicação para alguns destes passos..

Denomina-se:

$z_{\min}$ = melhor valor encontrado para a função objetivo durante o processo .

$k_{\min}$ = número da iteração para a qual foi encontrado $z_{\min}$, desde que o processo de memória de longo prazo foi solicitado.

$x_0 = x_{\text{best}}$ melhor solução desde o início da busca.

$z_0 = z_{\text{best}}$ = melhor valor encontrado para a função objetivo, desde que se iniciou a busca.

Algoritmo 6.1 - Algoritmo TS para o PTCF

Inicialização

Passo 0

A partir da solução básica factível inicial, computa-se o valor de Δ_{ij} para cada $(i,j) \notin \tau$ e introduz-se o arco (i,j) na árvore geradora se a variação líquida da função objetivo for $\Delta_{ij} < 0$. Se $\Delta_{ij} \geq 0$ para todo $(i,j) \notin \tau$ tem-se a **primeira solução ótima local**.

Para essa solução ótima local com valor da função objetivo igual a z , compute o parâmetro α .

Faça o contador de iteração $k = 0$;

Faça $l = 0$ e $l_1 = 0$

Para todo i e j

Faça $h_{ij} = 0$, $z_{ij} = z$ e $t_{ij} = -\infty$

Fim Para

Faça $i = 0$

Procedimento memória de curto prazo

Passo 1

Faça $i = i + 1$

Se $i > m$

Faça $i = 1$

Fim Se

Compute Δ_{iq} para todo $q=1$ até n e $(i,q) \notin \tau$

Passo 2

Selecione (i,j) como um arco entrando tal que:

$$\Delta_{ij} = \min\{\Delta_{iq} \mid (i,q) \notin \tau, 1 \leq q \leq n\},$$

com um arco saindo (i_s, j_s) .

Se $\Delta_{ij} = \infty$, vá para o **Passo 1**.

Passo 3

Se $(k - t_{ij}) \leq \xi_{in}$ ou $(k - t_{rs}) \leq \xi_{out}$

Vá para o **Passo 4**

Do contrário

Vá para o **Passo 5**

Fim Se

Passo 4

Se $z + \Delta_{ij} > z_{ij}$ ou $z + \Delta_{ij} > z_{rs}$

Faça $\Delta_{ij} = \infty$ e Vá para o **Passo 2**.

Do contrário

Vá para o **Passo 5**.

Fim Se

Passo 5

Atualize a árvore geradora.

Faça

$$h_{rs} = h_{rs} + (k - t_{rs})$$

$$t_{ij} = k$$

$$t_{rs} = k$$

$$z_{ij} = z$$

$$z_{rs} = z$$

Fim Faça

Faça

$$k = k + 1$$

$$z = z + \Delta_{ij}$$

Fim Faça

Passo 6

Se $z < z_{\min}$

Faça $z_{\min} = z$ e $k_{\min} = k$.

Se $z < z_0$

Faça $z_0 = z$ e Salve x_0 (solução corrente)

Fim Se

Fim Se

Passo 7

Se $(k - k_{\min}) < \beta N$

Vá para o **Passo 1** (continuar processo de memória de curto prazo).

Fim Se

Se $\beta N \leq (k - k_{\min}) < (\beta + \gamma)N$

Vá para o **Passo 8** (iniciar processo 1 de memória intermediária).

Do Contrário

Vá para o **Passo 12** (iniciar processo 2 de memória intermediária).

Fim Se

Processo 1 de memória intermediária

Passo 8

Faça $i = i + 1$

Se $i > m$

Faça $i = 1$

Fim Se

Para todo $q=1$ até n e com $(i,q) \notin \tau$

Compute $\Delta'_{iq} = \Delta_{iq} - \alpha \frac{h_{iq}}{k}$

Fim Para

Passo 9

Selecione (i,j) como um arco entrando tal que

$$\Delta'_{ij} = \min\{\Delta'_{iq} \mid (i,q) \notin \tau, 1 \leq q \leq n\},$$

com um arco saindo (i_s, j_s) .

Se $\Delta'_{ij} = \infty$, vá para o **Passo 8**.

Fim Se

Passo 10

Se $(k - t_{ij}) \leq \xi_{in}$ ou $(k - t_{rs}) \leq \xi_{out}$

Vá para o **Passo 11**

Do contrário

Vá para o **Passo 5**

Fim Se

Passo 11

Se $z + \Delta_{ij} > z_{ij}$ ou $z + \Delta_{ij} > z_{rs}$

Faça $\Delta'_{ij} = \infty$ e Vá para o **Passo 9**.

Do contrário

Vá para o **Passo 5**.

Fim Se

Processo 2 de memória intermediária

Passo 12

Faça $(i_o, j_o) = (i, n)$

Passo 13:

Faça $i = i + 1$

Se $i > m$

Faça $i = 1$

Fim Se

Fim Faça

Passo 14

Para $j = 1$ até n e $(i,j) \notin \tau$

(Repita Passos 14 (a) até 14 (d))

(a) Compute Δ_{ij}

(b) Se $\Delta_{ij} \geq 0$ e Se $(i,j) = (i_o, j_o)$

Vá para o **Passo 16**

(c) Se $\Delta_{ij} < 0$

Faça $(i_o, j_o) = (i, j)$

Faça (i,j) entrar para a árvore geradora com arco saindo (i_s, j_s)

Fim Se

(d) Atualize a árvore geradora

Faça

$$h_{rs} = h_{rs} + (k - t_{rs})$$

$$t_{ij} = k$$

$$t_{rs} = k$$

$$Z_{ij} = Z$$

$$Z_{rs} = Z$$

Fim Faça

Faça $k = k + 1$

Faça $z = z + \Delta_{ij}$

Passo 15

Vá para o **Passo 13**.

Conclusão dos processos de memória intermediária e regra de parada

Passo 16

Se $z < z_0$

Faça $z_0 = z$ e Salve x_0 (solução corrente)

Se $l \geq L$

STOP

Fim Se

Faça $i_0 = i$

Se $l_1 = 0$

Faça $l_1 = 1$

Vá para o **Passo 17**

Do Contrário

Faça $l_1 = 0$

Vá para o **Passo 23.**

Fim Se

Processo 1 de memória de longo prazo

Passo 17

Faça $i = i + 1$

Se $i > m$

Faça $i = 1$

Fim Se

Compute Δ''_{iq} para todo $q=1$ até n e $(i,q) \notin \tau$

Passo 18

Selecione (i_e, j_e) como um arco entrante tal que:

$$\Delta''_{ij} = \min \{ \Delta''_{iq} \mid (i, q) \notin \tau, 1 \leq q \leq n \},$$

com um arco saindo (i_s, j_s) .

Se $\Delta''_{ij} = \infty$, vá para o **Passo 17.**

Passo 19

Se $(k - t_{ij}) \leq \xi_{in}$ ou $(k - t_{rs}) \leq \xi_{out}$

Vá para o **Passo 20**

Do contrário

Vá para o **Passo 21**

Fim Se

Passo 20

Se $z + \Delta_{ij} > z_{ij}$ ou $z + \Delta_{ij} > z_{rs}$

Faça $\Delta''_{ij} = \infty$ e Vá para o **Passo 18**.

Do contrário

Vá para o **Passo 21**.

Fim Se

Passo 21

Atualize a árvore geradora.

Faça

$$h_{rs} = h_{rs} + (k - t_{rs})$$

$$t_{ij} = k$$

$$t_{rs} = k$$

$$z_{ij} = z$$

$$z_{rs} = z$$

Fim Faça

Faça

$$k = k + 1$$

$$z = z + \Delta_{ij}$$

Fim Faça

Passo 22

Se $i = i_0$

Faça $z_{\min} = z$ e $k_{\min} = k$.

Reinicialize os valores de ξ_{in} e ξ_{out}

Vá para o **Passo 1**

Do Contrário

Vá para o **Passo 17**

Fim Se

Processo 2 de memória de longo prazo

Passo 23

Faça $i = i + 1$

Se $i > m$

Faça $i = 1$

Fim Se

Selecione (i_e, j_e) como um arco entrando tal que

$$t_{ij} = \min \{t_{iq} \mid (i, q) \notin \tau, 1 \leq q \leq n\},$$

com um arco saindo (i_s, j_s) .

Compute Δ_{ij}

Passo 24

Atualize a árvore geradora.

Faça

$$h_{rs} = h_{rs} + (k - t_{rs})$$

$$t_{ij} = k$$

$$t_{rs} = k$$

$$z_{ij} = z$$

$$z_{rs} = z$$

Fim Faça

Faça

$$k = k + 1$$

$$z = z + \Delta_{ij}$$

Fim Faça

Passo 25

Se $i = i_o$

Faça $l = l + 1$.

Faça $z_{\min} = z$

Faça $k_{\min} = k$

Reinicialize os valores de ξ_{in} e ξ_{out}

Vá para o **Passo 1**

Do Contrário

Vá para o **Passo 23**.

FIM

No processo memória intermediária 2, passo 14, (i_o, j_o) é o último arco que foi introduzido na árvore geradora, após terem sido verificados todos os não básicos, uma a um. Se for encontrado um arco $(i_e, j_e) \notin \tau$ e com $\Delta_{ij} < 0$, ele é introduzido na árvore e (i_o, j_o) é atualizado.

Se o final da lista de arcos é atingido, ela é reiniciada do topo. Quando (i_o, j_o) é alcançado, sabe-se que não existem mais arcos $(i, j) \notin \tau$ com $\Delta_{ij} < 0$, e aquele é uma solução ótima local que foi alcançada.

No Passo 16, utiliza-se o ponteiro l_1 para se efetuar a mudança cíclica entre as duas estratégias do processo de memória de longo prazo 1 e 2.

O valor l do Passo 25 conta o número de vezes que o processo de memória de longo prazo é solicitado.

O i_o nos Passos 16, 22 e 25 armazena a última origem que foi verificada no processo de memória de longo prazo, reiniciando-se o próximo processo a partir desta linha.

CAPÍTULO 7

7. RESULTADOS

Descrevem-se abaixo os resultados obtidos na implementação efetuada neste trabalho, para os problemas obtidos nas categorias de problemas pequenos e fáceis, médios e grandes.

Utilizou-se o algoritmo PPL, baseado em BALINSKI (1961) para gerar a SBF inicial e o algoritmo TS, baseado em SUN et al. (1998), para a melhoria das soluções obtidas pelo primeiro e compararam-se os resultados obtidos, em termos percentuais, com o ótimo global fornecido.

Os problemas pequenos e fáceis foram obtidos da literatura consultada e de GOTTLIEB (2001) sendo os demais diretamente dele. Os problemas médios e grandes (50x100), gerados aleatoriamente são totalmente conectados e densos em termos de custos fixos. A quantidade fornecida/demandada para os problemas 50x100 é de 50.000. As soluções ótimas globais para os problemas médios e grandes também se encontram disponíveis em GOTTLIEB (2001), sendo que os problemas 50x100 foram os mesmos utilizados por SUN et al. (1998).

Foram utilizados 6 problemas na categoria de problemas pequenos e fáceis, 14 na categoria de problemas médios e 15 na categoria de problemas grandes. Os problemas médios e grandes foram gerados randomicamente e com custos inteiros. Os custos unitários variam na faixa de 3 a 8 e os custos fixos são de vários tipos.

O procedimento **TS** obteve o ótimo global de modo rápido, com um pequeno número de iterações, para os problemas pequenos e fáceis. Ficou bastante próximo do ótimo global para os problemas médios e ligeiramente acima do ótimo global no caso dos problemas grandes. No experimento limitou-se o número de chamadas às memórias de longo prazo 1 e 2 a apenas 2 de cada, através da fixação do parâmetro $L=lmax=4$.

De modo a obter um compromisso entre a qualidade das soluções e o tempo de resposta, procurou-se ajustar empiricamente, de modo adequado, os valores dos parâmetros β e γ . Estes parâmetros definem o número de iterações que o algoritmo TS vai realizar dentro de cada processo de, sem que haja variação no valor de $zmin$.



Os parâmetros β e γ , para os problemas pequenos, foram fixados em 0,30 e 0,70, respectivamente, enquanto que para os problemas médios, foram adotados os valores 0,05 e 0,70. Para os problemas grandes (50x100) foram utilizados os pares (0,05 e 0,10) e (0,05 e 0,15), para β e γ respectivamente, correspondendo os limites de 250 iterações (alterações da base), sem que se houvesse melhoria no valor z_{min} , para a busca local (memória de curto prazo). Têm-se as faixas 250-750 e 250-1000, respectivamente, como número de iterações dentro do processo de intensificação (memória intermediária 1) e, acima dos 750 e 1000, respectivamente, para a chamada da intensificação efetuada pela memória intermediária 2, que por sua vez aciona alternadamente os processos de diversificação chamados de memória de longo prazo 1 e 2.

Atribuíram-se valores para os parâmetros da lista tabu dentro das faixas: $7 \leq \xi_{in} \leq 12$ e $5 \leq \xi_{out} \leq 7$, para os problemas grandes, enquanto que para os problemas pequenos e médios atribuíram-se valores dentro das faixas $7 \leq \xi_{in} \leq 12$ e $2 \leq \xi_{out} \leq 4$.

O problema ran14x18 foi utilizado exaustivamente para se testar o algoritmo TS, durante a sua fase de implementação. Ele apresenta muitos pontos degenerados o que dificulta a busca, conforme observado na lista de discussão descrita na *home page FCTP instances* de GOTTLIEB (2001), onde se têm depoimentos da utilização de softwares comerciais como o CPLEX, XPRESS, PARINO, MINTO e LINGO para a sua resolução.

Para os problemas pequenos e simples o TS obteve o ótimo global (100,0%) em todos os casos; para os problemas médios as soluções obtidas pelo TS ficaram na faixa de 100,0% a 106,2% e com média de 101,7% em relação aos ótimos globais e, para os problemas 50x100, os ótimos locais ficaram na faixa de 100,9% a 103,7% e com média de 102,2%, em relação aos ótimos globais, podendo se considerar estes resultados como bastante satisfatórios.

Os resultados comparativos obtidos pelas diversas heurísticas estão apresentados nas tabelas 7.1 a 7.6, enquanto que no Anexo A, apresentam-se os problemas de testes pequenos/fáceis, no formato FCTP, bem como o formato do relatório de saída do TS, para a heurística de BALINSKI associada à regra do Custo Mínimo, associada à regra do Canto Noroeste e para o algoritmo TS..



Problemas Pequenos/Fáceis

Valores iniciais obtidos pela heurística de Balinski e programação linear											
para os problemas fáceis - exemplos dos livros e de gottlieb											
dados originais		p/balinski+regra noroeste+pl					p/balinski+regra custo min+pl				
problema	res.livros/got	o/c.fixo distr.		o/custo fixo	número de	%rel.ot.glob.	o/c.fixo distr.		o/custo fixo	número de	%rel.ot.glob.
FCIP	ótimo global	vr.inicial	vr.final	ótimo local	iterações	noro/ot.glob	vr.inicial	vr.final	ótimo local	iterações	omn/ot.glob
bk4x3	360,00	364,17	321,67	360,00	3	102,9%	370,00	321,67	360,00	3	102,9%
gr4x6	202,35	191,71	185,55	234,95	4	116,1%	196,27	185,55	234,95	4	116,1%
puc4x6	112,00	139,00	112,00	115,00	5	102,7%	112,00	112,00	112,00	0	100,0%
murty5x7	710,00	1228,00	627,50	826,00	7	116,3%	742,83	627,50	802,00	3	113,0%
bal8x12	471,55	483,39	451,19	504,55	10	107,0%	493,75	451,19	504,55	11	107,0%
kb2x3	65,00	64,80	64,67	65,00	1	100,0%	64,80	64,67	65,00	1	100,0%
médias	318,48	421,38	293,76	360,92	5,0	107,5%	329,94	293,76	346,42	3,7	106,5%

Tabela 7.1 - Resultados obtidos por Balinski e programação linear – fáceis - bk4x3: problema 4x3 de Balinski e Krarup ap. Gottlieb (2001); gr4x6: problema 4x6 de Gray ap. Gottlieb (2001); puc4x6: problema 4x6 de Puccini (1985); murty5x7: problema 5x7 de Murty (1968); bal8x12: problema 8x12 de Balinski ap. Gottlieb (2001); e kb2x3: problema 2x3 de Kuhn e Baumol ap. Salkin (1989).

Resultados obtidos pelo algoritmo TS										
para os problemas fáceis - exemplos dos livros e de gottlieb										
dados originais		p/balinski+regra noroeste+pl			p/balinski+regra custo min+pl			algoritmo TS		
problema	res.livros/got	ótimo local	número de	%rel.ot.glob.	ótimo local	número de	%rel.ot.glob.	ótimo local	número de	%rel.ot.glob.
FCTP	ótimo global	noro+PL	iterações	noro/ot.glob	omni+PL	iterações	omni/ot.glob	TS	iterações	TS/ot.glob
bk4x3	360,00	360,00	3	102,9%	360,00	3	102,9%	360,00	12	100,0%
gr4x6	202,35	234,95	4	116,1%	234,95	4	116,1%	202,35	9	100,0%
puc4x6	112,00	115,00	5	102,7%	112,00	0	100,0%	112,00	4	100,0%
murty5x7	710,00	826,00	7	116,3%	802,00	3	113,0%	710,00	59	100,0%
bal8x12	471,55	504,55	10	107,0%	504,55	11	107,0%	471,55	204	100,0%
kb2x3	65,00	65,00	1	100,0%	65,00	1	100,0%	65,00	7	100,0%
médias	318,48	360,92	5,0	107,5%	346,42	3,7	106,5%	318,48	49,2	100,0%

Tabela 7.2 - Resultados comparativos TS e Balinski – fáceis - bk4x3: problema 4x3 de Balinski e Krarup ap. Gottlieb (2001); gr4x6: problema 4x6 de Gray ap. Gottlieb (2001); puc4x6: problema 4x6 de Puccini (1985); murty5x7: problema 5x7 de Murty (1968); bal8x12: problema 8x12 de Balinski ap. Gottlieb (2001); e kb2x3: problema 2x3 de Kuhn e Baumol ap. Salkin (1989).

Problemas Médios

Valores iniciais obtidos pela heurística de Balinski e programação linear											
para os problemas de médio porte gerados randomicamente por Gottlieb.											
dados originais		p/balinski+regra noroeste+pl					p/balinski+regra custo min+pl				
problema	res. Gottlieb	c/c.fixo distr.		c/custo fixo	número de	% rel.gottlieb	c/c.fixo distr.		c/custo fixo	número de	% rel.gottlieb
FCTP	por B&B	vr.inicial	vr.final	ótimo local	iterações	noro/B&B	vr.inicial	vr.final	ótimo local	iterações	cmin/B&B
ran10x10a	1499,0	2387,5	1252,7	1616,0	22	107,8%	1481,6	1252,7	1736,0	15	115,8%
ran10x10b	3073,0	4214,1	2613,5	3337,0	19	108,6%	3341,1	2613,5	3337,0	15	108,6%
ran10x10c	13007,0	13248,3	11203,1	16636,0	18	127,9%	11442,9	11203,1	16636,0	10	127,9%
ran10x12	2714,0	4019,7	2426,2	3294,0	19	121,4%	2581,8	2426,2	3290,0	9	121,2%
ran10x26	4270,0	6468,6	3857,0	4782,0	47	112,0%	4609,1	3857,0	4782,0	25	112,0%
ran12x12	2291,0	3077,9	1826,6	2668,0	20	116,5%	1988,7	1826,6	2668,0	9	116,5%
ran12x21	3664,0	5998,6	3157,4	4158,0	46	113,5%	4010,6	3157,4	4098,0	21	111,8%
ran13x13	3252,0	4633,6	2691,4	3598,0	27	110,6%	3221,3	2691,4	3486,0	15	107,2%
ran14x18	3712,0	5187,6	3016,9	4282,0	45	115,4%	3492,1	3016,9	4347,0	19	117,1%
ran16x16	3823,0	5125,6	3116,4	4393,0	30	114,9%	3450,2	3116,4	4456,0	15	116,6%
ran17x17	1373,0	3015,5	1215,2	1454,0	48	105,9%	1569,3	1215,2	1514,0	26	110,3%
ran4x64	9711,0	14523,6	9637,9	10012,0	59	103,1%	12162,9	9637,9	10007,0	41	103,0%
ran6x43	6330,0	9793,8	6244,7	6691,0	44	105,7%	9032,4	6244,7	6645,0	36	105,0%
ran8x32	5247,0	8092,5	4937,6	5837,0	51	111,2%	6296,3	4937,5	5837,0	30	111,2%
médias	4569,0	6413,4	4085,5	5197,0	35	112,5%	4905,7	4085,5	5202,8	20	113,2%

Tabela 7.3 - Resultados obtidos por Balinski e programação linear – médios - ran10x10a a ran8x32: problemas fctp gerados randomicamente com dimensões 10x10 a 8x32 de Gottlieb (2001).

Resultados obtidos pelo algoritmo TS										
para os problemas de médio porte gerados randomicamente por Gottlieb.										
dados originais		p/balinski+regra noroeste+pl			p/balinski+regra custo min+pl			algoritmo TS		
problema	res. Sun	ótimo local	n.iter.	% rel.ót.glob.	ótimo local	n.iter.	% rel.ót.glob.	ótimo TS	n.iter.	% rel.ót.glob.
FCTP	ótimo global	noro+PL		noro/ót.glob.	cmin+PL		cmin/ót.glob.	alfa=.05/beta=.07		
ran10x10a	1499,0	1616,0	22	107,8%	1736,0	15	115,8%	1499,0	254	100,0%
ran10x10b	3073,0	3337,0	19	108,6%	3337,0	15	108,6%	3092,0	143	100,6%
ran10x10c	13007,0	16636,0	18	127,9%	16636,0	10	127,9%	13115,0	311	100,8%
ran10x12	2714,0	3294,0	19	121,4%	3290,0	9	121,2%	2881,0	409	106,2%
ran10x26	4270,0	4782,0	47	112,0%	4782,0	25	112,0%	4445,0	1009	104,1%
ran12x12	2291,0	2668,0	20	116,5%	2668,0	9	116,5%	2291,0	571	100,0%
ran12x21	3664,0	4158,0	46	113,5%	4098,0	21	111,8%	3726,0	1297	101,7%
ran13x13	3252,0	3598,0	27	110,6%	3486,0	15	107,2%	3284,0	959	101,0%
ran14x18	3712,0	4282,0	45	115,4%	4347,0	19	117,1%	3824,0	1063	103,0%
ran16x16	3823,0	4393,0	30	114,9%	4456,0	15	116,6%	3903,0	2988	102,1%
ran17x17	1373,0	1454,0	48	105,9%	1514,0	26	110,3%	1373,0	2004	100,0%
ran4x64	9711,0	10012,0	59	103,1%	10007,0	41	103,0%	9737,0	83	100,3%
ran6x43	6330,0	6691,0	44	105,7%	6645,0	36	105,0%	6381,0	180	100,8%
ran8x32	5247,0	5837,0	51	111,2%	5837,0	30	111,2%	5402,0	46	103,0%
médias	4569,0	5197,0	35	112,5%	5202,8	20	113,2%	4639,5	808	101,7%

Tabela 7.4 - Resultados comparativos TS e Balinski – médios - ran10x10a a ran8x32: problemas fctp gerados randomicamente com dimensões 10x10 a 8x32 de Gottlieb (2001).

Problemas Grandes

Valores iniciais obtidos pela heurística de Balinski e programação linear											
para os problemas de grande porte (50x100) gerados randomicamente por Sun et.al.											
dados originais		p/balinski+regra noroeste+pl					p/balinski+regra custo min+pl				
problema	res.Sun	c/c.fixo distr.		c/custo fixo	número de	% rel.sun	c/c.fixo distr.		c/custo fixo	número de	% rel.sun
FCTP	por B&B	vr.inicial	vr.final	ótimo local	iterações	noro/B&B	vr.inicial	vr.final	ótimo local	iterações	cmin/B&B
n3700	1254130,0	2141493,0	972305,4	1342264,0	418	107,0%	1348367,0	972305,9	1345704,0	209	107,3%
n3701	1244592,0	2028008,0	961764,0	1309690,0	403	105,2%	1370635,0	961764,1	1309295,0	212	105,2%
n3702	1227393,0	2007380,0	961184,3	1326705,0	389	108,1%	1333383,0	961184,3	1319875,0	196	107,5%
n3703	1216236,0	1962637,0	934212,1	1316163,0	383	108,2%	1193300,0	934212,3	1315936,0	208	108,2%
n3704	1244366,0	2158246,0	969800,4	1326602,0	421	106,6%	1219535,0	969800,4	1327781,0	202	106,7%
n3705	1229634,0	2099708,0	973361,0	1342738,0	396	109,2%	1300799,0	973361,0	1342279,0	219	109,2%
n3706	1241381,0	2042075,0	960882,1	1322093,0	358	106,5%	1293608,0	960882,1	1323586,0	204	106,6%
n3707	1203955,0	2046638,0	935136,8	1295125,0	398	107,6%	1284571,0	935136,8	1302405,0	225	108,2%
n3708	1224734,0	2166210,0	967522,9	1329354,0	406	108,5%	1274118,0	967522,9	1328949,0	205	108,5%
n3709	1216562,0	2170284,0	959314,3	1324453,0	392	108,9%	1271563,0	959314,1	1323031,0	192	108,8%
n370a	1272014,0	2073615,0	979219,8	1353827,0	377	106,4%	1286044,0	979219,5	1353827,0	227	106,4%
n370b	1243314,0	2198889,0	988308,4	1368880,0	380	110,1%	1308947,0	988308,3	1371613,0	204	110,3%
n370c	1228867,0	2106882,0	965430,8	1343329,0	391	109,3%	1322685,0	965430,9	1346041,0	197	109,5%
n370d	1228857,0	2106882,0	965430,8	1343329,0	391	109,3%	1322685,0	965430,9	1346041,0	197	109,5%
n370e	1220967,0	2135685,0	961651,3	1317883,0	373	107,9%	1275288,0	961651,3	1317883,0	191	107,9%
médias	1233133,5	2096308,8	963701,6	1330829,0	392	107,9%	1293701,9	963701,7	1331616,4	206	108,0%

Tabela 7.5 - Resultados obtidos por Balinski e programação linear – grandes n3700 a n370e: problemas fctp gerados randomicamente com dimensões 50x100 de Sun et.al ap. Gottlieb (2001).

Resultados obtidos pelo algoritmo TS													
para os problemas de grande porte (50x100) gerados randomicamente por Sun et.al.													
dados originais													
problema	res.Sun	ótimo local	n.iter.	% rel.ót.glob.	ótimo local	n.iter.	% rel.ót.glob.	ótimo TS	n.iter.	% rel.ót.glob.	ótimo TS	n.iter.	% rel.ót.glob.
FCTP	ótimo global	noro+PL		noro/ót.glob.	cmin+PL		cmin/ót.glob.	alfa=.05/beta=.15				alfa=.05/beta=.10	
n3700	1254130,0	1342264,0	418	107,0%	1345704,0	209	107,3%	1267873,0	6297	101,1%	1268113,0	2969	101,1%
n3701	1244592,0	1309690,0	403	105,2%	1312769,0	212	105,5%	1254920,0	4090	100,8%	1255938,0	3592	100,9%
n3702	1227393,0	1326705,0	389	108,1%	1319203,0	196	107,5%	1264496,0	3674	103,0%	1251438,0	3245	102,0%
n3703	1216236,0	1316163,0	383	108,2%	1315924,0	208	108,2%	1244701,0	3808	102,3%	1242460,0	3362	102,2%
n3704	1244366,0	1326602,0	421	106,6%	1327781,0	202	106,7%	1274900,0	3998	102,5%	1274669,0	3310	102,4%
n3705	1229634,0	1342738,0	396	109,2%	1342561,0	219	109,2%	1269059,0	3835	103,2%	1262234,0	3975	102,7%
n3706	1241381,0	1322093,0	358	106,5%	1323789,0	204	106,6%	1269987,0	3763	102,3%	1263750,0	3198	101,8%
n3707	1203955,0	1295125,0	398	107,6%	1302363,0	225	108,2%	1235274,0	3961	102,6%	1237320,0	3279	102,8%
n3708	1224734,0	1329354,0	406	108,5%	1329519,0	205	108,6%	1263576,0	3892	103,2%	1270001,0	6316	103,7%
n3709	1216562,0	1324453,0	392	108,9%	1325502,0	192	109,0%	1247976,0	3709	102,6%	1248356,0	3474	102,6%
n370a	1272014,0	1353827,0	377	106,4%	1353578,0	227	106,4%	1279901,0	4513	100,6%	1278813,0	6104	100,5%
n370b	1243314,0	1368880,0	380	110,1%	1370409,0	204	110,2%	1279472,0	3913	102,9%	1277482,0	5433	102,7%
n370c	1228867,0	1343329,0	391	109,3%	1346115,0	197	109,5%	1268640,0	4363	103,2%	1259892,0	4866	102,5%
n370d	1228857,0	1343329,0	391	109,3%	1346115,0	197	109,5%	1268640,0	4363	103,2%	1259892,0	4866	102,5%
n370e	1220967,0	1317883,0	373	107,9%	1319251,0	191	108,0%	1250571,0	4344	102,4%	1248565,0	5412	102,3%
médias	1233133,5	1330829,0	392	107,9%	1332038,9	206	108,0%	1262665,7	4168	102,4%	1259928,2	4227	102,2%

Tabela 7.6 - Resultados comparativos TS e Balinski – grandes - ran10x10a a ran8x32: problemas fctp gerados randomicamente com dimensões 10x10 a 8x32 de Gottlieb (2001).

CAPÍTULO 8

8. CONCLUSÕES

Os problemas de transporte com carga fixa são problemas de programação inteira mista porque algumas das variáveis de decisão são do tipo 0-1 envolvendo decisões do tipo sim-não, enquanto as outras variáveis são do tipo real ou mesmo inteira. Esses problemas são muito difíceis de se resolver principalmente quando se têm portes médios ou grandes e os algoritmos para eles desenvolvidos são geralmente muito pouco eficientes e dependentes do número de variáveis inteiras envolvidas e da estrutura especial do problema que deve ser explorada.

Para problemas de programação linear inteira mista tem sido utilizado com sucesso implementações do tipo *branch-and-bound*. [...] No caso dos problemas de transporte com carga fixa pode se preservar a estrutura de fluxo em rede do problema. GARFINKEL e NEMHAUSER (1972, p.389)

A aplicação do método simplex para o problema com custos modificados no caso do problema de transporte com carga fixa ou a simples aplicação do método simplex para o problema relaxado, em geral com o arredondamento da solução ótima obtida, pode levar a resultados muitos insatisfatórios principalmente quando as variáveis assumem pequenos valores.

No desenvolvimento do algoritmo PPL especializado para o problema de transporte, utilizou-se a heurística de BALINSKI (1961), gerando as SBF iniciais para os problemas de testes, utilizadas para a inicialização do algoritmo TS. Na implementação de BALINSKI adotou-se o critério de máxima variação da função objetivo para se os movimentos de uma configuração para outra, até se obter o ótimo local para o PTCF e, para os objetivos pretendidos, pode-se considerar que foram obtidos bons resultados.

Foi implementada apenas a fase I do algoritmo de WALKER, utilizando-se $m+n$ variáveis artificiais para gerar uma base para o problema, e por iterações de PPL, primeiro sem levar em conta os custos fixos, até que todas as variáveis artificiais tivessem sido eliminadas da base, gerando uma base para o algoritmo, aplicando então a fase I de WALKER, para se obter um ótimo local. Não se utilizou o



algoritmo para todos os problemas de teste, uma vez que ele não foi totalmente implementado, mas seria interessante que se efetuasse a sua implementação e aplicação para os problemas aqui utilizados e comparassem os resultados obtidos pelo algoritmo TS.

A aplicação de algoritmos heurísticos como o TS é uma abordagem que se torna atrativa, principalmente quando envolvem um grande número de variáveis, em virtude de o TS trabalhar de modo rápido e gerar boas soluções. Estas características são bastante interessantes, principalmente em se tratando de problemas de planejamento, quando os dados envolvidos são apenas hipóteses, com alguma margem de incerteza, a serem comprovados posteriormente durante e após a implantação do projeto, quando deverão ocorrer pequenos ajustes.

Pelos resultados obtidos constata-se que o TS funciona bem na maioria dos problemas de transporte com carga fixa testados embora seja altamente dependente dos parâmetros de controle que devem ser fixados após muitas tentativas, ou seja, de modo empírico, utilizando-se problemas com ótimos conhecidos e com dimensões equivalentes as dos problemas que se deseja resolver.

Em casos práticos onde não se conhece o ótimo global do problema, o ideal seria, além da fixação de modo empírico de faixas de valores para os parâmetros, implementar a alteração desses parâmetros de forma aleatória ao longo do processo de busca. Outro procedimento interessante é o de se executar o algoritmo um certo número de vezes para cada problema, tomando-se como ótimo local o melhor resultado entre os obtidos nos vários testes.

Constata-se que a Busca Tabu, com a implementação efetuada, consegue trabalhar adequadamente os conceitos de intensificação da busca e de diversificação da busca, conseguindo explorar adequadamente o espaço de soluções e encaminhar a busca para regiões fora dos ótimos locais que são obtidos ao longo dos processos, obtendo desse modo novos ótimos locais, porém altamente dependente dos parâmetros de controle fixados, exigindo muitas tentativas para se poder fixar os parâmetros de forma adequada.

As funções introduzidas para se incentivar certas configurações a entrar na base no processo de intensificação e para penalizar certas configurações em detrimento de outras soluções no processo diversificação, utilizando as informações armazenadas de modo implícito de ocorrências em termos de fatos recentes e de



freqüência com que certas configurações entram, saem ou permanecem na base funcionou adequadamente.

Pode-se concluir ainda, que na implementação efetuada, o TS se apresenta, a exemplo das técnicas que utilizam o PPL como base, como um algoritmo determinístico, ou seja, para determinados parâmetros de controle, chega-se às mesmas soluções, caminhando pelos mesmos ótimos locais, embora pudesse se dar um certo caráter aleatório ao TS pela implementação de alterações dos parâmetros de busca ao longo do processo, dentro de uma certa faixa de valores pré-determinados.



BIBLIOGRAFIA

- BALINSKI, M.L. *Fixed cost transportation problem*. Naval Research Logistics Quarterly 8, p.41-54, 1961.
- BARR, R.S., GLOVER, F., KLINGMAN, D. *A new optimization method for large scale fixed charge transportation problems*. Operations Research 29(3), p.448-463, 1981.
- BAZARAA, M.S., JARVIS, J.J., SHERALI, H.D. *Linear Programming and Network Flows*. 2th.ed. New York: John Wiley, 1990, p.419-527.
- BRONSON, R. *Pesquisa Operacional*. 1.ed. Trad: Bernardo Severo da Silva Filho, Othon Guilherme Pinto Bravo. São Paulo: McGraw-Hill do Brasil, 1985, p.52-96 e p.178-194.
- COOPER, L., DREBES, C. *An Aproximate Solution Method for the Fixed Charge Problem*. Naval Research Logistics Quarterly, 14, p.101-113, 1967.
- COOPER, L. *The Fixed Charge Problem: A New Heuristic Method*. Computer and Mathematics with Applications. 1(1), Pergamon Press, p.89-95, 1975.
- DANTZIG, G. *Professor George Dantzig: Linear Programming Founder Turns 80*. SIAM News, Nov.1994.
Disponível em : <http://www.stanford.edu/group/sol/dantzig.html>
Acesso em: 27 nov. 2001.
- DENZLER, D. *An Aproximate Algorithm for the Fixed Charge Problem*. Naval Research Logistics Quarterly 16, p.411-416, 1969.
- GARFINKEL, R.S. e NEMHAUSER, G.L. *Integer Programming*. New York: John Wiley, 1972, p.13-22, p.60-107 e p.385-390.
- GLOVER, F. Búsqueda Tabú. in; DÍAZ FERNÁNDEZ, A.(coord). *Optimización Heurística y Redes Neuronales*. Madrid: Paraninfo, 1996, p.105-142.
- GOLDBARG, M.C., LUNA, H.P.L. *Otimização Combinatória e Programação Linear: Modelos e Algoritmos*. Rio de Janeiro: Campus, 2000.
- GOTTLIEB, J, PAULMANN, L. *Genetic Algorithms for the Fixed Charge Transportation Problem*. Proceeding of the 5th IEEE International Conference on Evolutionary Computation, Anchorage, Alaska, p.330-335, 1998.
Disponível em: <http://www.in.tu-clausthal.de/~gottlieb/papers/icec98.abstract.html>
Acesso em: 11 mai. 2001.
- GOTTLIEB, J. *FCTP instances*.
Disponível em: <http://www.in.tu-clausthal.de/~gottlieb/benchmark/fctp/>
Acesso em: 11 mai. 2001.



HILLIER, F.S., LIEBERMAN, G.J. *Introduction to Operations Research*. 6th ed. New York: McGraw-Hill, 1995, p.303-352 e p.494-557.

_____. *Introdução à Pesquisa Operacional*. 1.ed.em português. Trad.de Helena L.Lemos da 3.ed. de *Introduction to Operations Research*. Holden-Day 1967. Rio de Janeiro: Campus./São Paulo: EDUSP, 1988, p.123-263 e p.694-724.

HIRSCH, W.M., DANTZIG, G.B. *The fixed charge problem*. Naval Research Logistics Quarterly 15, p.413-424, 1968.

HOUAISS, A. et al. *Dicionário Houaiss da língua portuguesa*. Elaborado no Instituto Antônio Houaiss de Lexicografia e Banco de Dados da Língua Portuguesa SC Ltda - Rio de Janeiro: Objetiva, 2001.

KUHN, H., BAUMOL, W. *An approximate Algorithm for the Fixed Charge Transportation Problem*. Naval Research Logistics Quartely, v.9, p.11-15, 1962.

MICHALEWICZ, Z. *Genetic Algorithms + Data Structures = Evolution Programs*. 3th ed. New York: Springer, 1996, p.181-208.

MURTY, K.G. *Solving the Fixed Charge Transportation Problem by Ranking the Extreme Points*. Operations Research 16(2), 1968, p.268-279.

_____. *Linear Programming*. New York: John Wiley, 1983.

PALEKAR, U.S., KARWAN, M.H., ZIONTS, S. *A Branch-and-Bound Method for the Fixed Charge Transportation Problem*. Management Sciences. 36(9), p.1092-1105, 1990.

PUCCINI, A.L. *Introdução à Programação Linear*. Rio de Janeiro: LTC, 1985, p.94-131.

GALLEGO RENDÓN, R.A. *Planejamento a Longo Prazo de Sistemas de Transmissão Usando Técnicas de Otimização Combinatorial*. Tese de Doutorado. Campinas: Unicamp, 1997, p.110-160.

SALKIN, H.M.F. *Integer Programming*. Wesley, 1975, p.482-504.

_____. *Foundations of Integer Programming*. Ney York: North-Holland, 1989.

STEINBERG, D.I. *The Fixed Charge Problem*. Nav.Res.Log.Quart. 17, p.217-236, 1970.

_____. *Designing a heuristic for the fixed-charge transportation problem*. Reprints in Mathematics and the Mathematical Sciences, Southern Illinois University at Edwardsville, Edwardsville, IL, 1977.

SUN, M., ARONSON, J.E., MCKEOWN, P.G., DRINKA, D. *A tabu search heuristic procedure for the fixed charge transportation problem*. European Journal of Operational Research 106, p.441-456, 1998.

TAHA, H.A. *Integer Programming: Theory, Applications, and Computations*. Orlando, Flórida: Academic Press, 1987, p.22-24 e p.285-304.

WALKER, W.E. *A heuristic adjacent extreme point algorithm for the fixed charge problem*. *Management Sciences* 22 (3), p.587-596, 1976.



ANEXO A

Problemas de Testes e Resultados: Pequenos/Fáceis



Problema bk4x3**Problema PTCF com 4 origens e 3 destinos
Balinski/Krarup, obtido de Gottlieb (2001).**

```
NAME : bk4x3
COMMENT : Beispiel von Balinski/Krarup
TYPE : FCTP
SOURCES : 4
DEMANDS : 3
SOURCE_SECTION
10.00
30.00
40.00
20.00
DEMAND_SECTION
20.00
50.00
30.00
VARCOST_SECTION
  2.00  3.00  4.00
  3.00  2.00  1.00
  1.00  4.00  3.00
  4.00  5.00  2.00
FIXCOST_SECTION
10.00 30.00 20.00
10.00 30.00 20.00
10.00 30.00 20.00
10.00 30.00 20.00
TRANSPORT_SECTION
  0.00  0.00 10.00
  0.00 30.00  0.00
20.00 20.00  0.00
  0.00  0.00 20.00
```

Problema bk4x3
Problema PTCF com 4 origens e 3 destinos
Balinski/Krarup, obtido de Gottlieb (2001).
Resolução como PPL por Balinski e Regra do Noroeste

matriz de custos balinski (unit+dist.fixos)

3.00	6.00	6.00
3.50	3.00	1.67
1.50	4.75	3.67
4.50	6.50	3.00

matriz da SBF solução básica factível inicial

10.00	-1.00	-1.00
10.00	20.00	-1.00
-1.00	30.00	10.00
-1.00	-1.00	20.00

SBF em termos de vetores

li	co	xb	cb	fb
1	1	10.00	3.00	10.00
2	1	10.00	3.50	10.00
2	2	20.00	3.00	30.00
3	2	30.00	4.75	30.00
3	3	10.00	3.67	20.00
4	3	20.00	3.00	20.00

valor inicial de $z_c =$ 364.166700

melhor solucao encontrada

-1.00	10.00	-1.00
-1.00	30.00	-1.00
20.00	10.00	10.00
-1.00	-1.00	20.00

final após 3 iterações

função objetivo de balinski = 321.666700
 função objetivo com custos fixos= 360.000000



Problema bk4x3
Problema PTCF com 4 origens e 3 destinos
Balinski/Krarup, obtido de Gottlieb (2001).
Resolução como PPL por Balinski e Regra do Custo Mínimo

matriz de custos balinski (unit+dist.fixos)

3.00	6.00	6.00
3.50	3.00	1.67
1.50	4.75	3.67
4.50	6.50	3.00

matriz da SBF solução básica factível inicial

-1.00	10.00	-1.00
-1.00	20.00	10.00
20.00	-1.00	20.00
-1.00	20.00	-1.00

SBF em termos de vetores

li	co	xb	cb	fb
1	2	10.00	6.00	30.00
2	2	20.00	3.00	30.00
2	3	10.00	1.67	20.00
3	1	20.00	1.50	10.00
3	3	20.00	3.67	20.00
4	2	20.00	6.50	30.00

valor inicial de z_c = 370.000000

melhor solucao encontrada

-1.00	10.00	-1.00
-1.00	30.00	-1.00
20.00	10.00	10.00
-1.00	-1.00	20.00

final após 3 iterações

função objetivo de balinski = 321.666700
 função objetivo com custos fixos= 360.000000

Problema bk4x3
Problema PTCF com 4 origens e 3 destinos
Balinski/Krarup, obtido de Gottlieb (2001).
Resolução pelo Algoritmo TS

matriz da SBF inicial

-1.0	10.0	-1.0
-1.0	30.0	-1.0
20.0	10.0	10.0
-1.0	-1.0	20.0

VALOR INICIAL DE z_c = 360.000000

melhor solucao encontrada

-1.0	-1.0	10.0
-1.0	30.0	-1.0
20.0	20.0	.0
-1.0	-1.0	20.0

após iterações = 12
com funcao objetivo atual= 380.000000
com ótimo obtido igual= 350.000000

parâmetros utilizados

nin =	7	nout=	2
beta	.30	gama=	.70
lim1=	3	lim2=	12



Problema gr4x6
Problema PTCF com 4 origens e 6 destinos
Gray, obtido de Gottlieb (2001).

NAME : gr4x6
COMMENT : Beispiel 2 von Gray
TYPE : FCTP
SOURCES : 4
DEMANDS : 6
SOURCE_SECTION
45.00
35.00
20.00
15.00
DEMAND_SECTION
35.00
30.00
25.00
15.00
5.00
5.00
VARCOST_SECTION
0.69 0.64 0.71 0.79 1.70 2.83
1.01 0.75 0.88 0.59 1.50 2.63
1.05 1.06 1.08 0.64 1.22 2.37
1.94 1.50 1.56 1.22 1.98 1.98
FIXCOST_SECTION
11.00 16.00 18.00 17.00 10.00 20.00
14.00 17.00 17.00 13.00 15.00 13.00
12.00 13.00 20.00 17.00 13.00 15.00
16.00 19.00 15.00 11.00 15.00 12.00

Problema gr4x6
Problema PTCF com 4 origens e 3 destinos
Gray, obtido de Gottlieb (2001).
Resolução como PPL por Balinski e Regra do Noroeste

matriz de custos balinski(unit+dist.fixos)

1.00	1.17	1.43	1.92	3.70	6.83
1.41	1.32	1.56	1.46	4.50	5.23
1.65	1.71	2.08	1.77	3.82	5.37
3.01	2.77	2.56	1.95	4.98	4.38

matriz da SBF solução básica factível inicial

35.00	10.00	-1.00	-1.00	-1.00	-1.00
-1.00	20.00	15.00	-1.00	-1.00	-1.00
-1.00	-1.00	10.00	10.00	-1.00	-1.00
-1.00	-1.00	-1.00	5.00	5.00	5.00

SBF em termos de vetores

li	co	xb	cb	fb
1	1	35.00	1.00	11.00
1	2	10.00	1.17	16.00
2	2	20.00	1.32	17.00
2	3	15.00	1.56	17.00
3	3	10.00	2.08	20.00
3	4	10.00	1.77	17.00
4	4	5.00	1.95	11.00
4	5	5.00	4.98	15.00
4	6	5.00	4.38	12.00

valor inicial de zc = 191.716700

melhor solucao encontrada

35.00	10.00	-1.00	-1.00	-1.00	-1.00
-1.00	20.00	15.00	-1.00	-1.00	-1.00
-1.00	-1.00	10.00	5.00	5.00	-1.00
-1.00	-1.00	-1.00	10.00	-1.00	5.00

final após 4 iterações
função objetivo de balinski = 185.550000
função objetivo com custos fixos= 234.950000

Problema gr4x6
Problema PTCF com 4 origens e 3 destinos
Gray, obtido de Gottlieb (2001).
Resolução como PPL por Balinski e Regra do Custo Mínimo

matriz de custos balinski(unit+dist.fixos)

1.00	1.17	1.43	1.92	3.70	6.83
1.41	1.32	1.56	1.46	4.50	5.23
1.65	1.71	2.08	1.77	3.82	5.37
3.01	2.77	2.56	1.95	4.98	4.38

matriz da SBF solução básica factível inicial

35.00	10.00	-1.00	-1.00	-1.00	-1.00
-1.00	20.00	.00	15.00	-1.00	-1.00
-1.00	-1.00	20.00	-1.00	-1.00	-1.00
-1.00	-1.00	5.00	-1.00	5.00	5.00

SBF em termos de vetores

li	co	xb	cb	fb
1	1	35.00	1.00	11.00
1	2	10.00	1.17	16.00
2	2	20.00	1.32	17.00
2	3	.00	1.56	17.00
2	4	15.00	1.46	13.00
3	3	20.00	2.08	20.00
4	3	5.00	2.56	15.00
4	5	5.00	4.98	15.00
4	6	5.00	4.38	12.00

valor inicial de zc = 196.266600

melhor solucao encontrada

35.00	10.00	-1.00	-1.00	-1.00	-1.00
-1.00	20.00	15.00	-1.00	-1.00	-1.00
-1.00	-1.00	10.00	5.00	5.00	-1.00
-1.00	-1.00	-1.00	10.00	-1.00	5.00

final após 4 iterações

função objetivo de balinski = 185.550000

função objetivo com custos fixos= 234.950000



Problema gr4x6
Problema PTCF com 4 origens e 3 destinos
Gray, obtido de Gottlieb (2001).
Resolução pelo Algoritmo TS

matriz da SBF inicial

35.0	10.0	-1.0	-1.0	-1.0	-1.0
-1.0	20.0	15.0	-1.0	-1.0	-1.0
-1.0	-1.0	10.0	5.0	5.0	-1.0
-1.0	-1.0	-1.0	10.0	-1.0	5.0

VALOR INICIAL DE z_c = 234.950000

melhor solucao encontrada

20.0	-1.0	25.0	-1.0	-1.0	-1.0
-1.0	30.0	.0	-1.0	-1.0	5.0
15.0	-1.0	-1.0	-1.0	5.0	-1.0'
.0	-1.0	-1.0	15.0	-1.0	-1.0

após iterações =	9
chamadas mem.lp =	0
com funcao objetivo atual=	211.750000
com ótimo obtido igual=	202.350000

parâmetros utilizados

nin =	10	nout=	3
beta	.30	gama=	.70
lim1=	7	lim2=	24



Problema puc4x6
Problema PT com 4 origens e 6 destinos
Puccini (1985)

```

NAME : puccini 4x6
COMMENT : exemplo p.116 livro prog.linear/puccini
TYPE : FCTP
SOURCES : 4
DEMANDS : 6
SOURCE_SECTION
5
6
2
9
DEMAND_SECTION
4
4
6
2
4
2
VARCOST_SECTION
9.0 12.0 9.0 6.0 9.0 10.0
7.0 3.0 7.0 7.0 5.0 5.0
6.0 5.0 9.0 11.0 3.0 11.0
6.0 8.0 11.0 2.0 2.0 10.0
FIXCOST_SECTION
0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0
TRANSPORT_SECTION
3.0 7.0 0.0 4.0 7.0 0.0
3.0 0.0 0.0 7.0 5.0 0.0
0.0 0.0 0.0 9.0 1.0 4.0
0.0 3.0 2.0 0.0 0.0 3.0

```

Problema puc4x6
Puccini (1985)
Resolução como PPL por Balinski e Regra do Noroeste

matriz de custos balinski (unit+dist.fixos)

9.00	12.00	9.00	6.00	9.00	10.00
7.00	3.00	7.00	7.00	5.00	5.00
6.00	5.00	9.00	11.00	3.00	11.00
6.00	8.00	11.00	2.00	2.00	10.00

matriz da SBF solução básica factível inicial

4.00	1.00	-1.00	-1.00	-1.00	-1.00
-1.00	3.00	3.00	-1.00	-1.00	-1.00
-1.00	-1.00	2.00	-1.00	-1.00	-1.00
-1.00	-1.00	1.00	2.00	4.00	2.00

SBF em termos de vetores

li	co	xb	cb	fb
1	1	4.00	9.00	.00
1	2	1.00	12.00	.00
2	2	3.00	3.00	.00
2	3	3.00	7.00	.00
3	3	2.00	9.00	.00
4	3	1.00	11.00	.00
4	4	2.00	2.00	.00
4	5	4.00	2.00	.00
4	6	2.00	10.00	.00

valor inicial de zc = 139.000000

melhor solucao encontrada

1.00	-1.00	4.00	-1.00	-1.00	-1.00
-1.00	4.00	-1.00	-1.00	-1.00	2.00
-1.00	.00	2.00	-1.00	-1.00	-1.00
3.00	-1.00	-1.00	2.00	4.00	-1.00

final após 5 iterações

função objetivo de balinski = 112.000000
função objetivo com custos fixos= 115.000000

Problema puc4x6
Puccini (1985)
Resolução como PPL por Balinski e Regra do Custo Mínimo

matriz de custos balinski(unit+dist.fixos)

9.00	12.00	9.00	6.00	9.00	10.00
7.00	3.00	7.00	7.00	5.00	5.00
6.00	5.00	9.00	11.00	3.00	11.00
6.00	8.00	11.00	2.00	2.00	10.00

matriz da SBF solução básica factível inicial

-1.00	-1.00	5.00	-1.00	-1.00	-1.00
-1.00	3.00	1.00	-1.00	-1.00	2.00
1.00	1.00	-1.00	-1.00	-1.00	-1.00
3.00	-1.00	-1.00	2.00	4.00	-1.00

SBF em termos de vetores

li	co	xb	cb	fb
1	3	5.00	9.00	.00
2	2	3.00	3.00	.00
2	3	1.00	7.00	.00
2	6	2.00	5.00	.00
3	1	1.00	6.00	.00
3	2	1.00	5.00	.00
4	1	3.00	6.00	.00
4	4	2.00	2.00	.00
4	5	4.00	2.00	.00

valor inicial de zc = 112.000000

melhor solucao encontrada

-1.00	-1.00	5.00	-1.00	-1.00	-1.00
-1.00	3.00	1.00	-1.00	-1.00	2.00
1.00	1.00	-1.00	-1.00	-1.00	-1.00
3.00	-1.00	-1.00	2.00	4.00	-1.00

final após 0 iteração
 função objetivo de balinski = 112.000000
 função objetivo com custos fixos= 112.000000



Problema puc4x6
Puccini (1985)
Resolução pelo Algoritmo TS

matriz da SBF inicial

-1.0	-1.0	5.0	-1.0	-1.0	-1.0
-1.0	3.0	1.0	-1.0	-1.0	2.0
1.0	1.0	-1.0	-1.0	-1.0	-1.0
3.0	-1.0	-1.0	2.0	4.0	-1.0

VALOR INICIAL DE z_c = 112.000000

melhor solucao encontrada

-1.0	-1.0	5.0	-1.0	-1.0	-1.0
-1.0	3.0	1.0	-1.0	-1.0	2.0
1.0	1.0	-1.0	-1.0	-1.0	-1.0
3.0	-1.0	-1.0	2.0	4.0	-1.0

após iterações = 4
com funcao objetivo atual= 120.000000
com ótimo obtido igual= 112.000000

parâmetros utilizados

nin =	7	nout=	2
beta	.30	gama=	.70
lim1=	7	lim2=	24



Problema murty5x7
Problema PTCF com 5 origens e 7 destinos
adaptado de Murty (1968)

NAME : murty5x7
COMMENT : exemplo livro murty /adaptado para carga fixa
TYPE : FCTP
SOURCES : 5
DEMANDS : 7
SOURCE_SECTION
6
4
3
2
2
DEMAND_SECTION
3
2
4
2
1
2
3
VARCOST_SECTION
3.0 12.0 11.0 2.0 15.0 16.0 21.0
9.0 13.0 19.0 11.0 10.0 6.0 17.0
13.0 15.0 3.0 16.0 9.0 12.0 18.0
5.0 6.0 17.0 13.0 18.0 11.0 22.0
8.0 12.0 14.0 4.0 13.0 7.0 23.0
FIXCOST_SECTION
30.0 120.0 110.0 20.0 150.0 160.0 210.0
90.0 130.0 190.0 110.0 100.0 60.0 170.0
130.0 150.0 30.0 160.0 90.0 120.0 180.0
50.0 60.0 170.0 130.0 180.0 110.0 220.0
80.0 120.0 140.0 40.0 130.0 70.0 230.0

Problema murty5x7
adaptado de Murty (1968)
Resolução como PPL por Balinski e Regra do Noroeste

matriz de custos balinski(unit+dist.fixos)

13.00	72.00	38.50	12.00	165.00	96.00	91.00
39.00	78.00	66.50	66.00	110.00	36.00	73.67
56.33	90.00	13.00	96.00	99.00	72.00	78.00
30.00	36.00	102.00	78.00	198.00	66.00	132.00
48.00	72.00	84.00	24.00	143.00	42.00	138.00

matriz da SBF solução básica factível inicial

3.00	2.00	1.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	3.00	1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	1.00	1.00	1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	1.00	1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	2.00

SBF em termos de vetores

li	co	xb	cb	fb
1	1	3.00	13.00	30.00
1	2	2.00	72.00	120.00
1	3	1.00	38.50	110.00
2	3	3.00	66.50	190.00
2	4	1.00	66.00	110.00
3	4	1.00	96.00	160.00
3	5	1.00	99.00	90.00
3	6	1.00	72.00	120.00
4	6	1.00	66.00	110.00
4	7	1.00	132.00	220.00
5	7	2.00	138.00	230.00

valor inicial de zc = 1228.000000

melhor solucao encontrada

3.00	-1.00	2.00	1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	1.00	-1.00	.00	3.00
-1.00	-1.00	2.00	-1.00	1.00	-1.00	-1.00
-1.00	2.00	-1.00	-1.00	-1.00	.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	2.00	-1.00

final após 7

função objetivo de balinski = 627.500000

função objetivo com custos fixos= 826.000000



Problema murty5x7
adaptado de Murty (1968)
Resolução como PPL por Balinski e Regra do Custo Mínimo

matriz de custos balinski(unit+dist.fixos)

13.00	72.00	38.50	12.00	165.00	96.00	91.00
39.00	78.00	66.50	66.00	110.00	36.00	73.67
56.33	90.00	13.00	96.00	99.00	72.00	78.00
30.00	36.00	102.00	78.00	198.00	66.00	132.00
48.00	72.00	84.00	24.00	143.00	42.00	138.00

matriz da SBF solução básica factível inicial

3.00	-1.00	1.00	2.00	-1.00	-1.00	-1.00
-1.00	-1.00	.00	-1.00	-1.00	2.00	2.00
-1.00	-1.00	3.00	-1.00	-1.00	-1.00	-1.00
-1.00	1.00	-1.00	-1.00	-1.00	-1.00	1.00
-1.00	1.00	-1.00	-1.00	1.00	-1.00	-1.00

SBF em termos de vetores

li	co	xb	cb	fb
1	1	3.00	13.00	30.00
1	3	1.00	38.50	110.00
1	4	2.00	12.00	20.00
2	3	.00	66.50	190.00
2	6	2.00	36.00	60.00
2	7	2.00	73.67	170.00
3	3	3.00	13.00	30.00
4	2	1.00	36.00	60.00
4	7	1.00	132.00	220.00
5	2	1.00	72.00	120.00
5	5	1.00	143.00	130.00

valor inicial de $z_c =$ 742.833300

melhor solucao encontrada

3.00	-1.00	1.00	2.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	1.00	3.00
-1.00	-1.00	3.00	-1.00	-1.00	-1.00	-1.00
-1.00	2.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	.00	-1.00	.00	1.00	1.00	-1.00

final após 3

função objetivo de balinski = 627.500000

função objetivo com custos fixos= 802.000000



Problema murty5x7
adaptado de Murty (1968)
Resolução pelo Algoritmo TS

matriz da SBF inicial

3.0	-1.0	1.0	2.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0	-1.0	1.0	3.0
-1.0	-1.0	3.0	-1.0	-1.0	-1.0	-1.0
-1.0	2.0	-1.0	-1.0	-1.0	-1.0	-1.0
-1.0	.0	-1.0	.0	1.0	1.0	-1.0

VALOR INICIAL DE z_c = 802.000000

melhor solucao encontrada

3.0	.0	1.0	2.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0	1.0	-1.0	3.0
-1.0	-1.0	3.0	-1.0	-1.0	-1.0	-1.0
-1.0	2.0	-1.0	-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	.0	.0	2.0	-1.0

após iterações = 17
 com funcao objetivo atual= 955.000000
 com ótimo obtido igual= 710.000000

parâmetros utilizados

nin =	10	nout=	3
beta	.30	gama=	.70
lim1=	10	lim2=	35



Problema bal8x12
Problema PTCF com 8 origens e 12 destinos
Balinski obtido de Gottlieb (2001)

NAME : bal8x12

COMMENT : Beispiel 1 von Balinski

TYPE : FCTP

SOURCES : 8

DEMANDS : 12

SOURCE_SECTION

15.00

20.00

45.00

35.00

25.00

35.00

10.00

25.00

DEMAND_SECTION

20.00

15.00

20.00

15.00

5.00

20.00

30.00

10.00

35.00

25.00

10.00

5.00

VARCOST_SECTION

0.69	0.64	0.71	0.79	1.70	2.83	2.02	5.64	5.94	5.94	5.94	7.68
------	------	------	------	------	------	------	------	------	------	------	------

1.01	0.75	0.88	0.59	1.50	2.63	2.26	5.64	5.85	5.62	5.85	4.94
------	------	------	------	------	------	------	------	------	------	------	------

1.05	1.06	1.08	0.64	1.22	2.37	1.66	5.64	5.91	5.62	5.91	4.94
------	------	------	------	------	------	------	------	------	------	------	------

1.94	1.50	1.56	1.22	1.98	1.98	1.36	6.99	6.99	6.99	6.99	3.68
------	------	------	------	------	------	------	------	------	------	------	------

1.61	1.40	1.61	1.33	1.68	2.83	1.54	4.26	4.26	4.26	4.26	2.99
------	------	------	------	------	------	------	------	------	------	------	------

5.29	5.94	6.08	5.29	5.96	6.77	5.08	0.31	0.21	0.17	0.31	1.53
------	------	------	------	------	------	------	------	------	------	------	------

5.29	5.94	6.08	5.29	5.96	6.77	5.08	0.55	0.35	0.40	0.19	1.53
------	------	------	------	------	------	------	------	------	------	------	------

5.29	6.08	6.08	5.29	5.96	6.45	5.08	2.43	2.30	2.33	1.81	2.50
------	------	------	------	------	------	------	------	------	------	------	------

FIXCOST_SECTION

11.0	16.0	18.0	17.0	10.0	20.0	17.0	13.0	15.0	12.0	14.0	14.0
------	------	------	------	------	------	------	------	------	------	------	------

14.0	17.0	17.0	13.0	15.0	13.0	16.0	11.0	20.0	11.0	15.0	10.0
------	------	------	------	------	------	------	------	------	------	------	------

12.0	13.0	20.0	17.0	13.0	15.0	16.0	13.0	12.0	13.0	10.0	18.0
------	------	------	------	------	------	------	------	------	------	------	------

16.0	19.0	16.0	11.0	15.0	12.0	18.0	12.0	18.0	13.0	13.0	14.0
------	------	------	------	------	------	------	------	------	------	------	------

19.0	18.0	15.0	16.0	12.0	14.0	20.0	19.0	11.0	17.0	16.0	18.0
------	------	------	------	------	------	------	------	------	------	------	------

13.0	20.0	20.0	17.0	15.0	12.0	14.0	11.0	12.0	19.0	15.0	16.0
------	------	------	------	------	------	------	------	------	------	------	------

11.0	12.0	15.0	10.0	17.0	11.0	11.0	16.0	10.0	18.0	17.0	12.0
------	------	------	------	------	------	------	------	------	------	------	------

17.0	10.0	20.0	12.0	17.0	20.0	16.0	15.0	10.0	12.0	16.0	18.0
------	------	------	------	------	------	------	------	------	------	------	------

TRANSPORT_SECTION

00.0	15.0	00.0	00.0	00.0	00.0	00.0	00.0	00.0	00.0	00.0	00.0
------	------	------	------	------	------	------	------	------	------	------	------

00.0	00.0	20.0	00.0	00.0	00.0	00.0	00.0	00.0	00.0	00.0	00.0
------	------	------	------	------	------	------	------	------	------	------	------

20.0	00.0	00.0	00.0	05.0	20.0	00.0	00.0	00.0	00.0	00.0	00.0
------	------	------	------	------	------	------	------	------	------	------	------

00.0	00.0	00.0	00.0	00.0	00.0	30.0	00.0	00.0	00.0	00.0	05.0
------	------	------	------	------	------	------	------	------	------	------	------

00.0	00.0	00.0	15.0	00.0	00.0	00.0	00.0	00.0	00.0	10.0	00.0
------	------	------	------	------	------	------	------	------	------	------	------

00.0	00.0	00.0	00.0	00.0	00.0	00.0	00.0	35.0	00.0	00.0	00.0
------	------	------	------	------	------	------	------	------	------	------	------

00.0	00.0	00.0	00.0	00.0	00.0	00.0	10.0	00.0	00.0	00.0	00.0
------	------	------	------	------	------	------	------	------	------	------	------

00.0	00.0	00.0	00.0	00.0	00.0	00.0	00.0	00.0	25.0	00.0	00.0
------	------	------	------	------	------	------	------	------	------	------	------

Problema bal8x12
Problema PTCF com 8 origens e 12 destinos
Balinski obtido de Gottlieb (2001)
Resolução como PPL por Balinski e Regra do Noroeste

matriz de custos balinski(unit+dist.fixos)

1.42	1.71	1.91	1.92	3.70	4.16	3.15	6.94	6.94	6.74	7.34	10.48
1.71	1.88	1.73	1.46	4.50	3.28	3.06	6.74	6.85	6.17	7.35	6.94
1.65	1.93	2.08	1.77	3.82	3.12	2.19	6.94	6.25	6.14	6.91	8.54
2.74	2.77	2.36	1.95	4.98	2.58	1.96	8.19	7.50	7.51	8.29	6.48
2.56	2.60	2.36	2.40	4.08	3.53	2.34	6.16	4.70	4.94	5.86	6.59
5.94	7.27	7.08	6.42	8.96	7.37	5.55	1.41	.55	.93	1.81	4.73
6.39	7.14	7.58	6.29	9.36	7.87	6.18	2.15	1.35	2.20	1.89	3.93
6.14	6.75	7.08	6.09	9.36	7.45	5.72	3.93	2.70	2.81	3.41	6.10

matriz da SBF solução básica factível inicial

15.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
5.00	15.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	.00	20.00	15.00	5.00	5.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	15.00	20.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	10.00	10.00	5.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	30.00	5.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	10.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	10.00	10.00	5.00

SBF em termos de vetores

li	co	xb	cb	fb
1	1	15.00	1.42	11.00
2	1	5.00	1.71	14.00
2	2	15.00	1.88	17.00
3	2	.00	1.93	13.00
3	3	20.00	2.08	20.00
3	4	15.00	1.77	17.00
3	5	5.00	3.82	13.00
3	6	5.00	3.12	15.00
4	6	15.00	2.58	12.00
4	7	20.00	1.96	18.00
5	7	10.00	2.34	20.00
5	8	10.00	6.16	19.00
5	9	5.00	4.70	11.00
6	9	30.00	.55	12.00
6	10	5.00	.93	19.00
7	10	10.00	2.20	18.00
8	10	10.00	2.81	12.00
8	11	10.00	3.41	16.00
8	12	5.00	6.10	18.00

valor inicial de zc = 483.385700

melhor solucao encontrada

15.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	20.00	.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
5.00	15.00	-1.00	15.00	5.00	-1.00	5.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	20.00	15.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	10.00	-1.00	-1.00	10.00	-1.00	-1.00	5.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	10.00	25.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	.00	-1.00	-1.00	10.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	25.00	.00	-1.00	-1.00

final após	10 iterações	
função objetivo de balinski	=	451.188100
função objetivo com custos fixos	=	504.550000

Problema bal8x12
Problema PTCF com 8 origens e 12 destinos
Balinski obtido de Gottlieb (2001)
Resolução como PPL por Balinski e Regra do Custo Mínimo

matriz de custos balinski(unit+dist.fixos)

1.42	1.71	1.91	1.92	3.70	4.16	3.15	6.94	6.94	6.74	7.34	10.48
1.71	1.88	1.73	1.46	4.50	3.28	3.06	6.74	6.85	6.17	7.35	6.94
1.65	1.93	2.08	1.77	3.82	3.12	2.19	6.94	6.25	6.14	6.91	8.54
2.74	2.77	2.36	1.95	4.98	2.58	1.96	8.19	7.50	7.51	8.29	6.48
2.56	2.60	2.36	2.40	4.08	3.53	2.34	6.16	4.70	4.94	5.86	6.59
5.94	7.27	7.08	6.42	8.96	7.37	5.55	1.41	.55	.93	1.81	4.73
6.39	7.14	7.58	6.29	9.36	7.87	6.18	2.15	1.35	2.20	1.89	3.93
6.14	6.75	7.08	6.09	9.36	7.45	5.72	3.93	2.70	2.81	3.41	6.10

matriz da SBF solução básica factível inicial

15.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
5.00	-1.00	15.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
.00	15.00	-1.00	-1.00	5.00	20.00	-1.00	-1.00	-1.00	-1.00	-1.00	5.00
-1.00	-1.00	5.00	15.00	-1.00	-1.00	15.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	15.00	10.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	35.00	.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	10.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	.00	-1.00	15.00	10.00	-1.00

SBF em termos de vetores

li	co	xb	cb	fb
1	1	15.00	1.42	11.00
2	1	5.00	1.71	14.00
2	3	15.00	1.73	17.00
3	1	.00	1.65	12.00
3	2	15.00	1.93	13.00
3	5	5.00	3.82	13.00
3	6	20.00	3.12	15.00
3	12	5.00	8.54	18.00
4	3	5.00	2.36	16.00
4	4	15.00	1.95	11.00
4	7	15.00	1.96	18.00
5	7	15.00	2.34	20.00
5	8	10.00	6.16	19.00
6	9	35.00	.55	12.00
6	10	.00	.93	19.00
7	10	10.00	2.20	18.00
8	8	.00	3.93	15.00
8	10	15.00	2.81	12.00
8	11	10.00	3.41	16.00

valor inicial de zc = 493.750000

melhor solucao encontrada

15.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	20.00	.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
5.00	15.00	-1.00	15.00	5.00	-1.00	5.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	20.00	15.00	-1.00	-1.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	10.00	-1.00	10.00	-1.00	-1.00	5.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	10.00	25.00	-1.00	-1.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	.00	-1.00	-1.00	10.00	-1.00
-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	25.00	.00	-1.00

final após 11 iterações

função objetivo de balinski	=	451.188100
função objetivo com custos fixos=		504.550000



Problema bal8x12
Problema PTCF com 8 origens e 12 destinos
Balinski obtido de Gottlieb (2001)
Resolução pelo Algoritmo TS

matriz da SBF inicial

```

15.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0
-1.0 -1.0 20.0 .0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0
 5.0 15.0 -1.0 15.0  5.0 -1.0  5.0 -1.0 -1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -1.0 -1.0 -1.0 20.0 15.0 -1.0 -1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -1.0 -1.0 -1.0 -1.0 10.0 -1.0 10.0 -1.0 -1.0  5.0
-1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 10.0 25.0 -1.0 -1.0 -1.0
-1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0  .0 -1.0 -1.0 10.0 -1.0
-1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 25.0  .0 -1.0

```

VALOR INICIAL DE z_c = 504.550000

melhor solucao encontrada

```

-1.0  15.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0
-1.0    .0 20.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0
20.0    .0 -1.0 -1.0  5.0 20.0 -1.0 -1.0 -1.0 -1.0 -1.0  .0
-1.0 -1.0 -1.0 -1.0 -1.0 -1.0 30.0 -1.0 -1.0 -1.0 -1.0  5.0
 .0 -1.0 -1.0 15.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 10.0 -1.0
-1.0  .0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 35.0 -1.0 -1.0 -1.0
 .0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 10.0 -1.0 -1.0 -1.0 -1.0
 .0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 25.0 -1.0 -1.0

```

após iterações = 204
com funcao objetivo atual= 569.000000
com ótimo obtido igual= 471.550000

parâmetros utilizados
nin = 10 nout= 3
beta .30 gama= .70
lim1= 28 lim2= 96

Problema kb2x3**Problema PT com 2 origens e 3 destinos
Kuhn e Baumol obtido de SALKIN (1989)**

NAME : kb2x3
COMMENT : Kuhn/Baumol
TYPE : FCTP
SOURCES : 2
DEMANDS : 3
SOURCE_SECTION
9.00
7.00
DEMAND_SECTION
8.00
5.00
3.00
VARCOST_SECTION
6.00 5.00 3.00
3.00 2.00 1.00
FIXCOST_SECTION
0.00 0.00 2.00
3.00 1.00 0.00
TRANSPORT_SECTION
8.00 1.00 0.00
0.00 4.00 3.00



Problema kb2x3
Problema PT com 2 origens e 3 destinos
Kuhn e Baumol obtido de SALKIN (1989)
Resolução como PPL por Balinski e Regra do Noroeste

matriz de custos balinski(unit+dist.fixos)

6.00	5.00	3.67
3.43	2.20	1.00

matriz da SBF solução básica factível inicial

8.00	1.00	-1.00
-1.00	4.00	3.00

SBF em termos de vetores

li	co	xb	cb	fb
1	1	8.00	6.00	.00
1	2	1.00	5.00	.00
2	2	4.00	2.20	1.00
2	3	3.00	1.00	.00

valor inicial de z_c = 64.800000

melhor solucao encontrada

8.00	1.00	-1.00
-1.00	4.00	3.00

final após 1 iteração

função objetivo de balinski = 64.666670

função objetivo com custos fixos= 65.000000

Problema kb2x3
Problema PT com 2 origens e 3 destinos
Kuhn e Baumol obtido de SALKIN (1989)
Resolução como PPL por Balinski e Regra do Custo Mínimo

matriz de custos balinski (unit+dist.fixos)

6.00	5.00	3.67
3.43	2.20	1.00

matriz da SBF solução básica factível inicial

8.00	1.00	-1.00
-1.00	4.00	3.00

SBF em termos de vetores

li	co	xb	cb	fb
1	1	8.00	6.00	.00
1	2	1.00	5.00	.00
2	2	4.00	2.20	1.00
2	3	3.00	1.00	.00

valor inicial de $z_c =$ 64.800000

melhor solucao encontrada

8.00	1.00	-1.00
-1.00	4.00	3.00

final após 1 iteração

função objetivo de balinski	=	64.666670
função objetivo com custos fixos	=	65.000000



Problema kb2x3
Problema PT com 2 origens e 3 destinos
Kuhn e Baumol obtido de SALKIN (1989)
Resolução pelo Algoritmo TS

matriz da SBF inicial

8.0	1.0	-1.0
-1.0	4.0	3.0

VALOR INICIAL DE z_c = 65.000000

melhor solucao encontrada

8.0	1.0	-1.0
-1.0	4.0	3.0

após iterações = 7
 com funcao objetivo atual= 66.000000
 com ótimo obtido igual= 65.000000

parâmetros utilizados
 nin = 7 nout= 2
 beta .30 gama= .70
 lim1= 1 lim2= 6



UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"

Câmpus de Ilha Solteira
Programa de Pós-Graduação em Engenharia Elétrica
Av. Brasil Centro, 56
15385-000 Ilha Solteira - SP
www.dee.feis.unesp.br

