GUSTAVO VALENTIM LOCH

UMA NOVA ABORDAGEM NO PROCESSO ITERATIVO DE MELHORIA DE SOLUÇÃO NA RESOLUÇÃO DO PROBLEMA DE TRANSPORTE

Tese apresentada ao Curso de Pós-Graduação em Métodos Numéricos em Engenharia, Área de Concentração em Programação Matemática, Setores de Tecnologia e Ciências Exatas, Universidade Federal do Paraná, como requisito parcial à obtenção do título de Doutor em Ciências.

Orientador: Prof. Dr. Arinei Carlos Lindbeck da Silva

CURITIBA
2014

Aos meus pais, por todo
amor, sacrifícios e incentivo

AGRADECIMENTOS

Aos meus pais, João Matias e Valdeci, pelo apoio na realização deste trabalho e em todas as etapas da minha vida.

As minhas irmãs, Juliana e Carolina, pelos bons momentos de convivência.

Ao meu orientador, professor Arinei Carlos Lindbeck da Silva, pela disposição, pela atenção e pelos ensinamentos.

Aos professores do PPGMNE, por todos os ensinamentos e colaborações. Agradeço em especial os quais tive oportunidade de ser aluno: Ademir, Celso, Neida, Anselmo, Maria Terezinha, Arinei, Jair e Vomir.

A secretária Maristela Bandil, pela disposição e pela alegria demonstrada em todos os momentos. Ao secretário Jair por conseguir esclarecer as dúvidas

Aos meus colegas de curso, em especial Monica Beltrami e Cassius Scarpin, pela amizade e pelo companheirismo.

A CAPES, pelo auxílio financeiro durante o primeiro ano do doutorado.

A Deus, pela vida e proteção

A todos, que direta ou indiretamente contribuíram para a realização desse trabalho.

Na verdade só sabemos quão
pouco sabemos – com o saber
cresce a dúvida

GOETHE (1826)

RESUMO

Dentre os problemas de Pesquisa Operacional, o Problema de Transporte (PT) é destacado como um dos mais importantes, devido a sua estrutura especial e, principalmente, pelas aplicações que não se limitam a problemas de distribuição. Para a resolução do Problema de Transporte, é amplamente conhecido na literatura e utilizado o método MODI, no qual são calculadas as variáveis duais e com base nelas recalculados os valores dos custos atualizados e que apresenta economia de tempo para resolução em relação ao método *Stepping Stone*. Na presente tese, a resolução do PT pelo método MODI foi realizada por uma implementação utilizando estrutura de quadro para armazenamento de informações e outra de árvore, sendo concluído que a resolução em árvore gerou uma economia média de 60,24% de tempo em relação à resolução em quadro. Foi demonstrado, sem a utilização das variáveis duais, que é possível o recálculo dos custos atualizados somente em função dos custos atualizados da iteração anterior e com a implementação deste resultado houve redução de 80,78% no tempo de resolução em relação à implementação do método MODI em árvore. Desta forma, a redução média da implementação em árvore recalculando somente os custos atualizados necessários foi de 92,34% em relação à implementação em quadro. Para reduzir ainda mais o tempo de resolução foi proposta uma nova forma de critério para escolha da variável não básica a entrar na base, utilizando uma lista, denominada *ReferenciaCAN*, menor de variáveis candidatas a tornarem-se básica na iteração e também um parâmetro, denominado *PercentualEconomiaAnterior*, para evitar a seleção de variáveis não básicas que não gerassem uma economia unitária menor que a desejada. Com isso foi possível reduzir o tempo médio de resolução em 37,06% em relação a situação anterior. De forma final, o tempo de resolução para o método e implementação final proposto na presente tese obteve uma redução de 95,18% em relação ao tempo médio da implementação clássica do método MODI em quadro.

Palavras-chave: Problema de Transporte, método MODI, melhoria de solução, recálculo de custos atualizados, implementação computacional.

ABSTRACT

Among the Operational Research problems, the Transportation Problem (TP) is highlighted as one of the most important, due to its special structure, and especially by applications that are not limited to distribution problems. In order to solve the Transportation Problem, it is widely known in the literature and used the MODI method, in which the dual variables are calculated and based on them the reduced costs are recalculated, providing time saving when compared to the Stepping Stone method. In this thesis, the PT solver by MODI method was performed by using an implementation in the tableau structure for storing information and other using tree structure. It was concluded that when the problems were solved using tree structure it was generated an average savings of 60.24% of time in comparison to tableau structure. Therefore, it was demonstrated without the use of the dual variables, that it is possible recalculation of reduced costs only considering reduced costs of the previous iteration and the implementation of this rule resulted in 80.78% reduction in time to solve when compared to the implementation of the method MODI using tree structure. Thus, the average reduction in tree implementation recalculating only the updated costs required was 92.34% in relation to the implementation in tableau structure. In order to reduce again the time a new way has been proposed as criteria for the choice of the non basic variable to enter the basis, using a list, called *ReferenciaCAN*, with fewer variables candidates to become basic at current iteration and also a parameter, called *PercentualEconomiaAnterior*, to avoid the selection of non-basic variables that do not generate a smaller unitary economy than desired. It was then possible to reduce the average resolution time by 37.06% compared to the previous situation. After the modifications, the solver time for the final implementation and method proposed in this thesis achieved a reduction of 95.18% compared to the average time the classical implementation of MODI method in tableau structure.

Keywords: Transportation Problem, MODI method, solution improvement, reduced costs computing, computational implementation.

LISTA DE FIGURAS

FIGURA 1 – EXEMPLO DE GRAFO E DÍGRAFO 30
FIGURA 2 – EXEMPLOS DE FLORESTAS E ÁRVORE 32
FIGURA 3 – EXEMPLOS DE ÁRVORES ENRAIZADAS.....33
FIGURA 4 – REPRESENTAÇÃO DE UM GRAFO BIPARTIDO $K3,2$ 35
FIGURA 5 – EXEMPLO DE PRESENTAÇÃO DAS ÁRVORES GERADORAS PARA UM GRAFO BIPARTIDO $K3,2$ 36
FIGURA 6 – MATRIZ DE ADJACÊNCIA PARA O GRAFO DE ÁRVORES DA FIGURA 5.....36
FIGURA 7 – EXEMPLO DE ÁRVORE GERADORA A SER UTILIZADA PARA A CODIFICAÇÃO DE PRÜFER.....39
FIGURA 8 – EXEMPLO PARA A EXPLICAÇÃO DE TROCA DE RAIZ EM ÁRVORES GERADORAS 44
FIGURA 9 – REPRESENTAÇÃO GENÉRICA DO QUADRO DE TRANSPORTE....48
FIGURA 10 – EXEMPLO DE PROBLEMA DE TRANSPORTE REPRESENTADO EM QUADRO 48
FIGURA 11 – REPRESENTAÇÃO DO PROBLEMA DO TRANSPORTE EM GRAFO 49
FIGURA 12 – EXEMPLO DE SOLUÇÃO BÁSICA EM GRAFO PARA O PT.....49
FIGURA 13 – EXEMPLO DE ÁRVORE ENRAIZADA PARA O PROBLEMA DE TRANSPORTE.....50
FIGURA 14 – EXEMPLO DE SOLUÇÃO PARA UM PT EM QUE É NECESSÁRIO COMPLETAR A BASE COM VARIÁVEIS DEGENERADAS 54
FIGURA 15 – REPRESENTAÇÃO DA FLORESTA ANTES DA BASE SER COMPLETADA 55
FIGURA 16 – TRANSFORMAÇÃO DA FLORESTA EM ÁRVORE PELA INSERÇÃO DE VARIÁVEIS BÁSICAS DEGENERADAS 55
FIGURA 17 – SOLUÇÃO EM QUADRO APÓS A INSERÇÃO DAS VARIÁVEIS BÁSICAS DEGENERADAS 56
FIGURA 18 – EXEMPLO PARA EXPLICAÇÃO DO SIGNIFICADO DO CUSTO ATUALIZADO.....57

FIGURA 19 – EXEMPLO PARA ENCONTRAR O $\theta - loop$ EM GRAFO 61
FIGURA 20 – EXEMPLO DO $\theta - loop$ EM QUADRO.....62
FIGURA 21 – EXEMPLO DE UM GRAFO BIPARTIDO $K4,2$.....63
FIGURA 22 – ENUMERAÇÃO DAS ÁRVORES GERADORAS PARA UM GRAFO $K4,2$.....64
FIGURA 23 – MATRIZ DE ADJACÊNCIA PARA AS ÁRVORES GERADORAS DA FIGURA 22.....65
FIGURA 24 – EXEMPLO DE CAMINHO ENTRE ÁRVORES ADJACENTES PARA O PROBLEMA DE TRANSPORTE.....66
FIGURA 25 – PARTE DE MATRIZ DE DISTÂNCIA ENTRE ÁRVORES DO PROBLEMA DE TRANSPORTE PARA AS ÁRVORES GERADORAS DA FIGURA 22.....66
FIGURA 26 – EXEMPLO DE ÁRVORE PARA O PROBLEMA DO TRANSPORTE .71
FIGURA 27 – EXEMPLO DE CÁLCULOS DAS VARIÁVEIS DUAIS POR MEIO DE UMA BUSCA EM LARGURA NA ÁRVORE 72
FIGURA 28 – EXEMPLOS DE POSSIBILIDADE PARA $h*$ PARA A ENTRADA DE $2,1$ NA BASE.....73
FIGURA 29 – EXEMPLOS DE POSSIBILIDADE PARA $h*$ PARA A ENTRADA DE $2,6$ NA BASE.....74
FIGURA 30 – DIVISÃO DA ÁRVORE $\mathcal{T}$ DA FIGURA 26 NAS SUBÁRVORES $\mathcal{H}$ E $\mathcal{T} - \mathcal{H}$ 74
FIGURA 31 – SUBÁRVORE $\mathcal{H}$ PARA O EXEMPLO DA FIGURA 33, COM $p,q = O2,D6$ e $rs,ts = O3,D4$ 75
FIGURA 32 – SUBÁRVORE $\mathcal{T} - \mathcal{H}$ PARA O EXEMPLO DA FIGURA 29 (b), COM $p,q = O2,D6$ e $rs,ts = O3,D4$ 75
FIGURA 33 – REPRESENTAÇÃO DA SUBÁRVORE $\mathcal{H}'$ 76
FIGURA 34 – REPRESENTAÇÃO DA ÁRVORE $\mathcal{T}'$ 76
FIGURA 35 – EXEMPLO DE SUBCAMINHO NO $\theta - loop$.....79
FIGURA 36 – REPRESENTAÇÃO DE $\mathcal{T}$ E $\mathcal{T}'$ PARA O CASO 3.1 82
FIGURA 37 – REPRESENTAÇÃO DE $\mathcal{T}$ E $\mathcal{T}'$ PARA O CASO 3.2 85
FIGURA 38 – REPRESENTAÇÃO DE $\mathcal{T}$ E $\mathcal{T}'$ PARA O CASO 4.1 88
FIGURA 39 – REPRESENTAÇÃO DE $\mathcal{T}$ E $\mathcal{T}'$ PARA O CASO 4.2 90

FIGURA 40 – EXEMPLO DE VALORES PARA A VARIÁVEL ***ProblemaTransporte*** DO TIPO **strTransporte** 96
FIGURA 41 – EXEMPLO DE UMA VARIÁVEL DO TIPO **srtBasicas** NO PROBLEMA DE TRANSPORTE.....97
FIGURA 42 – EXEMPLO DE UMA VARIÁVEL DO TIPO **srtNaoBasicas** NO PROBLEMA DE TRANSPORTE.....98
FIGURA 43 – REPRESENTAÇÃO DA ÁRVORE GERADORA PARA A SOLUÇÃO BÁSICA FACTÍVEL DA FIGURA 40 99
FIGURA 44 – ESTRUTURA **strNoArvore** PARA ARMAZENAMENTO DAS INFORMAÇÕES DAS VARIÁVEIS DA ÁRVORE GERADORA REFERENTE A SOLUÇÃO BÁSICA ATUAL NO PROBLEMA DE TRANSPORTE.....100
FIGURA 45 – EXEMPLO DE VARIÁVEL DO TIPO **strNoArvore** PARA ARMAZENAMENTO DE INFORMAÇÕES REFRENTES AOS NÓS DE ORIGEM.....100
FIGURA 46 – EXEMPLO DE VARIÁVEL DO TIPO **strNoArvore** PARA ARMAZENAMENTO DE INFORMAÇÕES REFRENTES AOS NÓS DE DESTINO 101
FIGURA 47 – EXEMPLO DE PREENCHIMENTO DA VARIÁVEL $MatrizReferencia(,)$.....102
FIGURA 48 – VARIÁVEL ***Linha()*** APÓS A MUDANÇA DE ÁRVORE CAUSADA PELA TROCA DE BASE.....114
FIGURA 49 – VARIÁVEL ***Coluna()*** APÓS A MUDANÇA DE ÁRVORE CAUSADA PELA TROCA DE BASE.....114
FIGURA 50 – ÁRVORE GERADORA ASSOCIADA À NOVA SOLUÇÃO 115
FIGURA 51 – NOVOS VALORES DA VARIÁVEL ***Basicas()*** APÓS ATUALIZAÇÃO DAS QUANTIDADES GERADA PELA TROCA DE BASE.....116
FIGURA 52 – NOVOS VALORES DA VARIÁVEL ***NaoBasicas()*** APÓS ATUALIZAÇÃO POR CAUSA DA TROCA DE BASE 117
FIGURA 53 – VARIÁVEL $MatrizReferencia(,)$ APÓS O FIM DA ITERAÇÃO.....117
FIGURA 54 – EXEMPLO DA VARIÁVEL ***NaoBasicas()*** PARA A CRIAÇÃO DA LISTA *ReferenciaCAN* 121
FIGURA 55 – EXEMPLO DA VARIÁVEL ***NaoBasicas()*** PARA APÓS A TROCA DE BASE 123

FIGURA 56 – SITUAÇÃO DA ÁRVORE GERADORA PARA O PIOR CASO PARA IDENTIFICAÇÃO DO $\theta - loop$ 132
FIGURA 57 – REPRESENTAÇÃO EM REDE DO PROBLEMA DE TRANSPORTE COM CUSTO FIXO 172
FIGURA 58 – PROPOSTA DE REPRESENTAÇÃO EM QUADRO PARA UM PROBLEMA DE TRANSPORTE COM CUSTO FIXO 173
FIGURA 59 – EXEMPLO PARA A PROPOSTA DE REPRESENTAÇÃO EM QUADRO PARA UM PROBLEMA DE TRANSPORTE COM CUSTO FIXO 173
FIGURA 60 – EXEMPLO DE QUADRO PARA UMA SOLUÇÃO DO PROBLEMA DE TRANSPORTE COM CUSTO FIXO 174
FIGURA 61 – EXEMPLO DE ÁRVORE PARA UMA SOLUÇÃO DO PROBLEMA DE TRANSPORTE COM CUSTO FIXO 175
FIGURA 62 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A PRIMEIRA ITERAÇÃO 176
FIGURA 63 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A PRIMEIRA ITERAÇÃO 177
FIGURA 64 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A SEGUNDA ITERAÇÃO 177
FIGURA 65 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A SEGUNDA ITERAÇÃO 178
FIGURA 66 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A TERCEIRA ITERAÇÃO 179
FIGURA 67 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A TERCEIRA ITERAÇÃO 179
FIGURA 68 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A QUARTA ITERAÇÃO 180
FIGURA 69 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A QUARTA ITERAÇÃO 180
FIGURA 70 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A QUINTA ITERAÇÃO 181
FIGURA 71 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A QUINTA ITERAÇÃO 181

FIGURA 72 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A SEXTA ITERAÇÃO.....182
FIGURA 73 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A SEXTA ITERAÇÃO.....182

LISTA DE GRÁFICOS

GRÁFICO 1 – EXEMPLO DE EVOLUÇÃO DOS VALORES DE CUSTOS ATUALIZADOS UTILIZADOS NUM EXEMPLO 400X400 COM A LISTA *ReferenciaCAN* 122
GRÁFICO 2 – COMPARAÇÃO DO PERCENTUAL DE ECONOMIA DE TEMPO DA RESOLUÇÃO EM ÁRVORE EM RELAÇÃO À RESOLUÇÃO EM QUADRO PARA O PROBLEMA DE TRANSPORTE.....129
GRÁFICO 3 – COMPARAÇÃO DO PERCENTUAL DE ECONOMIA DE TEMPO DA RESOLUÇÃO EM ÁRVORE EM RELAÇÃO À RESOLUÇÃO EM QUADRO PARA PROBLEMAS DE DESIGNAÇÃO 131
GRÁFICO 4 – ECONOMIA DE TEMPO DE RESOLUÇÃO PARA PROBLEMAS DE TRANSPORTE DO A_Otimizado EM RELAÇÃO AO A_MODI 135
GRÁFICO 5 – ECONOMIA DE TEMPO DE RESOLUÇÃO PARA PROBLEMAS DE DESIGNAÇÃO DO A_Otimizado EM RELAÇÃO AO A_MODI 137
GRÁFICO 6 – NÚMERO DE VEZES QUE A LISTA FOI REINICIALIZADA PARA A RESOLUÇÃO DO PROBLEMA DE TRANSPORTE.....145
GRÁFICO 7 – ECONOMIA DO TEMPO DE RESOLUÇÃO GERADA PELA UTILIZAÇÃO DA LISTA *ReferenciaCAN* NOS PROBLEMAS DE DESIGNAÇÃO.....146
GRÁFICO 8 – COMPARAÇÃO DO NÚMERO DE VEZES QUE A LISTA ReferenciaCAN É ATUALIZADA PARA PROBLEMAS DE TRANSPORTE E DESIGNAÇÃO 147
GRÁFICO 9 – TEMPO DE RESOLUÇÃO DOS PROBLEMAS DE TRANSPORTE PARA DIFERENTES VALORES DE *PercentualEconomiaAnterior*149
GRÁFICO 10 – ECONOMIA DE TEMPO DE RESOLUÇÃO EM RELAÇÃO À *PercentualEconomiaAnterior*=1 149
GRÁFICO 11 – ECONOMIA DE TEMPO DE Lista0,79 EM RELAÇÃO À A_Otimizado EM FUNÇÃO DO NÚMERO DE DESTINOS PARA 10.000 NÓS DE ORIGEM 154

LISTA DE QUADROS

QUADRO 1 – ALGORITMO PARA REPRESENTAR UMA ÁRVORE GERADORA COMO UM CÓDIGO DE PRÜFER 39
QUADRO 2 – PASSO A PASSO DE UM EXEMPLO PARA CODIFICAÇÃO DE PRÜFER DE UMA ÁRVORE GERADORA.....40
QUADRO 3 – ALGORITMO PARA REPRESENTAÇÃO GRÁFICA DE UMA ÁRVORE GERADORA A PARTIR DE UM CÓDIGO DE PRÜFER. 40
QUADRO 4 – PASSO A PASSO DE UM EXEMPLO PARA REPRESENTAÇÃO GRÁFICA DE UMA ÁRVORE GERADORA A PARTIR DE UM CÓDIGO DE PRÜFER.....42
QUADRO 5 – ESTRUTURA PARA ARMAZENAMENTO DE INFORMAÇÕES DE UMA ÁRVORE GERADORA 43
QUADRO 6 – EXEMPLO DE REPRESENTAÇÃO DE UMA ÁRVORE NA ESTRUTURA DE Valiente (2010).....43
QUADRO 7 – INFORMAÇÕES DE PAI E FILHOS DE CADA VÉRTICE DA ÁRVORE GERADORA DA FIGURA 8 (a).....45
QUADRO 8– INFORMAÇÕES DE PAI E FILHOS DE CADA VÉRTICE DA ÁRVORE GERADORA DA FIGURA 8 (c).....45
QUADRO 9 – ESTRUTUTA **strTransporte** PARA ARMAZENAMENTO DAS INFORMAÇÕES DE UM PROBLEMA DE TRANSPORTE.....95
QUADRO 10 – ESTRUTURA **strBasicas** PARA ARMAZENAMENTO DAS INFORMAÇÕES DAS VARIÁVEIS BÁSICAS DA SOLUÇÃO ATUAL NO PROBLEMA DE TRANSPORTE 97
QUADRO 11 – ESTRUTURA **strNaoBasicas** PARA ARMAZENAMENTO DAS INFORMAÇÕES DAS VARIÁVEIS BÁSICAS DA SOLUÇÃO ATUAL NO PROBLEMA DE TRANSPORTE 98
QUADRO 12 – PROCEDIMENTO GERAL PARA RESOLUÇÃO DO PROBLEMA DE TRANSPORTE 103
QUADRO 13 – PROCEDIMENTO PARA PREENCHIMENTO INICIAL DA VARIÁVEL ***Basicas()*** E DE PARTE DA VARIÁVEL $MatrizReferencia(,)$.....104
QUADRO 14 – PROCEDIMENTO PARA PREENCHIMENTO INICIAL DA VARIÁVEL ***NaoBásicas()*** E DE PARTE DA VARIÁVEL $MatrizReferencia(,)$104

QUADRO 15 – PROCEDIMENTO PARA PREENCHIMENTO DE INFORMAÇÕES NAS VARIÁVEIS **Linha()** e **Coluna()** PARA REPRESENTAÇÃO DA ÁRVORE GERADORA ASSOCIADA A SOLUÇÃO BÁSICA FACTÍVEL.....105
QUADRO 16 – PROCEDIMENTO PARA O CÁLCULO DAS VARIÁVEIS DUAIS COM BASE NAS VARIÁVEL ***Basicas()*** JÁ ESTANDO PREENCHIDA 106
QUADRO 17 - FUNÇÃO PARA IDENTIFICAR QUAL A VARIÁVEL NÃO BÁSICA QUE APRESENTA O CUSTO ATUALIZADO MAIS NEGATIVO...106
QUADRO 18 – PROCEDIMENTO RESPONSÁVEL POR CRIAR A LISTA DE ANCESTRAIS DE UM NÓ ORIGEM E DE UM NÓ DESTINO.....107
QUADRO 19 – PROCEDIMENTO PARA ENCONTRAR POSICAO DO ANCESTRAL COMUM DE MAIOR PROFUNDIDADE.....108
QUADRO 20 – FUNÇÃO EncontraQuebra 109
QUADRO 21 – PROCEDIMENTO MudaArvoreOtimizada.....111
QUADRO 22 – PROCEDIMENTO AtualizaQuantidades 112
QUADRO 23 – EXEMPLO PARA AS LINHAS 06 A 24 DO CÓDIGO DO PROCEDIMENTO MudaArvoreOtimizada 112
QUADRO 24 – PROCEDIMENTO DeterminaArvoreH.....118
QUADRO 25 – PROCEDIMENTO AtualizaCustosAtualizados 119
QUADRO 27 – PROCEDIMENTO MudaArvore.....120
QUADRO 28 – PROCEDIMENTO ResolveProblemaTransporteListaCAN.....124
QUADRO 29 – ADAPTAÇÃO DA ESTRUTURA strTransporte PARA O PTCF.....161
QUADRO 30 – ADAPTAÇÃO DA ESTRUTURA strBasicas PARA O PTCF.....162
QUADRO 31 – ADAPTAÇÃO DA ESTRUTURA strNaoBasicas PARA O PTCF....163
QUADRO 32 164
QUADRO 33 – DESCRIÇÃO DA META-HEURÍSTICA ILS 183

LISTA DE TABELAS

TABELA 1 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO PARA AS ESTRUTURAS EM QUADRO E EM ÁRVORE 128
TABELA 2 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO PARA AS ESTRUTURAS EM QUADRO E EM ÁRVORE 130
TABELA 3 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO PARA A_MODI E A_Otimizado 135
TABELA 4 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO ENTRE MudaArvore e MudaArvoreOtimizada PARA O PROBLEMA DE DESIGNAÇÃO .136
TABELA 5 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO PARA A_Otimizado e A_MODI 139
TABELA 6 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO PARA AS ESTRUTURAS EM QUADRO E EM ÁRVORE 140
TABELA 7 – QUANTIDADE E PERCENTUAL DE CUSTOS ATUALIZADOS RECALCULADOS PARA OS PROBLEMAS DE TRANSPORTE ..141
TABELA 8 – QUANTIDADE E PERCENTUAL DE CUSTOS ATUALIZADOS RECALCULADOS PARA OS PROBLEMAS DE DESIGNAÇÃO ... 141
TABELA 9 – NÚMERO MÉDIO POR ITERAÇÃO DE ORIGENS E DESTIONS EM $\mathcal{H}$ 142
TABELA 10 – NÚMERO MÉDIO DE ELEMENTOS NAS LISTAS AncestraisOrigem E AncestraisDestino E VALOR MÉDIO DE PontoEncontro PARA OS PROBLEMAS DA SEÇÃO 5.3 143
TABELA 11 – COMPARAÇÃO DE TEMPOS E ITERAÇÕES PARA O PROBLEMA DE TRANSPORTE GERADAS PELA UTILIZAÇÃO DA LISTA *ReferenciaCAN* 144
TABELA 12 – COMPARAÇÃO DE TEMPOS E ITERAÇÕES PARA O PROBLEMA DE DESIGNAÇÃO GERADAS PELA UTILIZAÇÃO DA LISTA *ReferenciaCAN* 146
TABELA 13 – TEMPO DE RESOLUÇÃO PARA PROBLEMAS DE TRANSPORTE UTILIZANDO O PARÂMETRO *PercentualEconomiaAnterior* 148
TABELA 14 – TEMPO DE RESOLUÇÃO PARA PROBLEMAS DE DESIGNAÇÃO UTILIZANDO O PARÂMETRO PercentualEconomiaAnterior 151

TABELA 15 – DESEMPENHO DA UTILIZAÇÃO DA LISTA *ReferenciaCAN* E DO PARÂMETRO *PercentualEconomiaAnterior*=0,79 para PROBLEMAS DE DESIGNAÇÃO COM $m \gg n$ 153
TABELA 16 – TEMPOS MÉDIOS DE RESOLUÇÃO EM MILISSEGUNDOS DE PROBLEMAS DE TRANSPORTE PELO MÉTODO A_MODI 155
TABELA 17 – ECONOMIA DE TEMPO DE RESOLUÇÃO PELO MÉTODO A_Otimizado EM RELAÇÃO AO MÉTODO A_MODI 155
TABELA 18 – ECONOMIA DE TEMPO DE RESOLUÇÃO PELO MÉTODO LISTA_0,79 EM RELAÇÃO AO MÉTODO A_Otimizado 156
TABELA 19 – ECONOMIA DE TEMPO DE RESOLUÇÃO PELO MÉTODO LISTA_0,79 EM RELAÇÃO AO MÉTODO A_MODI 156
TABELA 20 – NÚMERO MÉDIO DE LISTAS UTILIZADAS NO MÉTODO LISTA_0,79 157
TABELA 21 – NÚMERO MÉDIO DE ITERAÇÕES POR LISTA NO MÉTODO LISTA_0,79 157

SUMÁRIO

1 INTRODUÇÃO 21
1.1 OBJETIVO GERAL 26
1.2 OBJETIVOS ESPECÍFICOS 26
1.3 RELEVÂNCIA E IMPORTÂNCIA 27
1.4 ESTRUTURA DO TRABALHO 27
2 REVISÃO DE LITERATURA 29
2.1 CONCEITOS DE GRAFOS 29
2.2 COMPLEMENTOS SOBRE ÁRVORES GERADORAS 34
2.2.1 Distância entre árvores geradoras 35
2.2.2 Árvore geradora central 37
2.2.3 Máxima distância entre árvores geradoras 38
2.2.4 Estrutura de dados para armazenamento da árvore 38
2.2.5 Troca de raiz de uma árvore geradora 44
2.3 CARACTERÍSTICA DO PROBLEMA DE TRANSPORTE 46
2.3.1 Existência de solução ótima com $m + n - 1$ variáveis básicas 46
2.3.2 Solução com valores inteiros e vantagens computacionais 47
2.3.3 Representação em quadro e árvore para o problema do transporte 47
2.3.4 Resolução do Problema de Transporte 51
2.3.5 Obtenção da solução básica factível inicial 52
2.3.6 Variáveis básicas degeneradas no Problema de Transporte 53
2.3.7 O significado do custo atualizado 56
2.3.8 O dual do Problema do Transporte 59
2.3.9 Mudança de solução básica 60
2.4 GRAFOS BIPARTIDOS, PROBLEMA DE TRANSPORTE E ÁRVORES GERADORAS 62
2.5 NOTA SOBRE COMPLEXIDADE DE RESOLUÇÃO DO PROBLEMA DE TRANSPORTE 68
3 RECÁLCULO DOS CUSTOS ATUALIZADOS NA MUDANÇA DE BASE 70
3.1 CÁLCULO DOS NOVOS CUSTOS ATUALIZADOS APÓS A ITERAÇÃO 70
4 IMPLEMENTAÇÃO COMPUTACIONAL 94
4.1 AS ESTRUTURAS DE DADOS UTILIZADAS 94

4.1.1 Estrutura de dados para representação do Problema de Transporte 94
4.1.2 Estrutura de dados para representação das variáveis básicas e não básicas..96
4.1.3 Estrutura de dados para representação da árvore geradora 99
4.1.4 Estrutura de dados para associação de variáveis.....101
4.2 PROCEDIMENTOS PARA A IMPLEMENTAÇÃO COMPUTACIONAL DO MÉTODO 102
4.2.1 Pseudocódigo geral para resolução do Problema de Transporte 103
4.2.2 Atualização da árvore geradora.....110
4.3 NOVO CRITÉRIO PARA ESCOLHA DA VARIÁVEL A TORNAR-SE BÁSICA.120
5 RESULTADOS COMPUTACIONAIS 126
5.1 COMPARAÇÃO ENTRE UTILIZAÇÃO DE ESTRUTURAS.....126
5.1.1 Comparação entre utilização de estruturas para Problemas de Transporte ...127
5.1.2 Comparação entre utilização de estruturas para Problemas de Designação .129
5.1.3 Operações necessárias para a identificação do $\theta - loop$ na estrutura de árvore 132
5.2 Comparação entre os recálculos dos custos atualizados para problemas com variáveis do tipo inteira 134
5.2.1 Comparação entre o recálculo dos custos atualizados para o Problema de Transporte para variáveis do tipo inteira.....134
5.2.2 Comparação entre o recálculo dos custos atualizados para o Problema de Designação para variáveis do tipo inteira 136
5.3 Comparação entre os recálculos dos custos atualizados para problemas com variáveis do tipo ponto flutuante de simples precisão.....138
5.3.1 Comparação entre o recálculo dos custos atualizados para o Problema de Transporte para variáveis do tipo ponto flutuante de simples precisão 138
5.3.2 Comparação entre o recálculo dos custos atualizados para o Problema de Designação para variáveis do tipo ponto flutuante de simples precisão.....139
5.3.3 Análise sobre o número de recálculos de custos atualizados para A_Otimizado 140
5.3.4 Análise sobre o $\theta - loop$ 142
5.4 DESEMPENHO PARA A UTILIZAÇÃO DA LISTA *ReferenciaCAN*.....143
5.4.1 Desempenho para a utilização da lista *ReferenciaCAN* para Problemas de Transporte 144
5.4.2 Desempenho para a utilização da lista *ReferenciaCAN* para Problemas de Designação.....145
5.5 INCLUSÃO DO PARÂMETRO *PercentualEconomiaAnterior* 147

5.5.1 Resultados para o parâmetro *PercentualEconomiaAnterior* nos Problemas de Transporte 148
5.5.2 Resultados para o parâmetro *PercentualEconomiaAnterior* nos Problemas de Designação 151
5.6 ANÁLISE PARA PROBLEMAS DE DIMENSÕES MAIORES 152
5.7 ANÁLISE PARA PROBLEMAS DE DESIGNAÇÃO COM $m \gg n$ 152
5.8 ANÁLISE FINAL PARA COMPARAÇÃO ENTRE A_MODI, A_Otimizado E LISTA *ReferenciaCAN* COM O PARÂMETRO *PercentualEconomiaAnterior* 154
6 CONSIDERAÇÕES FINAIS E SUGESTÕES PARA TRABALHOS FUTUROS 159
REFERÊNCIAS 165
APÊNDICE 172

1 INTRODUÇÃO

A área de Pesquisa Operacional (PO) surgiu, segundo Hillier e Lieberman (2010), nos primórdios da Segunda Guerra Mundial, por volta de 1940, com estudos para utilizar radares para interceptar aviões inimigos. Na época da guerra, a necessidade de trabalhar com recursos escassos de forma mais eficiente contribuiu para o aumento dos estudos científicos e, consequentemente, as pesquisas começaram a se expandir para problemas de operações, tais como: manutenção e inspeção de aviões, decisão de tipos de aviões para missões, dimensionamento de comboios de frota, etc. (ARENALES *et al*., 2007).

Diante do sucesso obtido pela aplicação de técnicas da PO no âmbito militar, foi natural a expansão do estudo para outras áreas. Em 1947, o Pentágono[1] implantou o projeto SCOOP (Computação Científica de Programação Ótima, do inglês *Scientific Computation of Optimal Programs*) com o objetivo de obter ferramentas de auxílio às decisões da força aérea americana. George Dantzig participava do projeto e desenvolveu, formalizou e testou o método simplex para resolver problemas de programação linear. Mais tarde, em conferências foi possível perceber que surgiam diferentes focos de estudo como: teorias de estoques, substituição de equipamentos, teoria de filas, programação *(scheduling*) de tarefas em máquinas, teoria de jogos, fluxos em redes e otimização linear. (ARENALES *et al*., 2007)

Durante as décadas seguintes, diversos problemas de programação linear que pudessem ser resolvidos pelo método simplex foram propostos e a evolução do poder de processamento dos computadores fez com que fosse possível encontrar soluções para problemas de tamanhos cada vez maiores.

Ao mesmo tempo, surgiram pesquisas para o desenvolvimento de métodos exatos[2] específicos com resolução em tempo polinomial para diversos problemas utilizando vantagens das particularidades de estrutura de cada um deles, como os algoritmos de Prim (1957), Kruskal (1956) e Baruvka (1926) para a árvore geradora

[1] Pentágono é a sede do Departamento de Defesa dos Estados Unidos, construído durante a segunda guerra mundial.

[2] Métodos exatos são aqueles que após a execução de uma sequência de passos previamente definidos, a solução encontrada é um ótimo global para o problema.

mínima, Floyd (1962) e Dijkstra (1959) para caminho mínimo entre dois pontos, Ford e Fulkerson (1956) para o problema de máximo fluxo.

O´Connor (2002) relata que o Problema de Transporte foi inicialmente estudado pelo matemático russo Kantorovich em 1939, mas o trabalho foi ignorado pelos russos e permaneceu desconhecido até 1960. Anos mais tarde, em 1975, os estudos de Kantorovich que também abordavam os problemas de distribuição de trabalho entre máquinas, distribuição de pedidos entre empresas, distribuição de materiais básicos, combustíveis e fatores de produção, minimização de desperdício em corte e plano ótimo para remessas de mercadorias lhe rendeu, em conjunto com Koopmans, o prêmio Nobel de economia.

No ocidente, o Problema de Transporte foi apresentado inicialmente por Hitchcock em 1941, que também esboçou um procedimento construtivo para resolvê-lo, similar ao método Simplex. De forma independente, Koopmans (1947) igualmente estudou este problema, o que fez com que o problema também viesse a ser chamado de Problema de Transporte Hitchcock-Koopmans (FORD e FULKERSON, 1956).

Segundo Silva (2012), a formulação matemática do Problema de Transporte já é conhecida desde 1941 e sua resolução por meio de algoritmo específico desde 1951.

O problema clássico de transporte pode ser expresso como sendo a minimização dos custos de transportar alguma *commodity*[3] de m origens (fontes, ofertas) para n destinos (sumidouros, demandas) existindo restrições de a_i suprimentos disponíveis na origem i e demandas b_j exigidas no destino j.

$$\min z(x) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \quad (1)$$

$$\sum_{j=1}^{n} x_{ij} = a_i, i = 1, \dots, m \quad (2)$$

$$\sum_{i=1}^{m} x_{ij} = b_j, j = 1, \dots, n \quad (3)$$

$$x_{ij} \geq 0, i = 1, \dots, m; j = 1, \dots, n \quad (4)$$

[3] *Commodities* são produtos padronizados que não possuem diferenciação. No caso do Problema do Transporte, isso significa que os produtos ofertados em cada origem são indiferentes.

A função objetivo (1) refere-se a minimização do custo total de transporte. O conjunto de restrições (2) impõe que toda a oferta de cada origem seja transportada. O conjunto de restrições (3) define que todas as demandas devem ser atendidas. Por fim (4) refere-se a não-negatividade das quantidades transportadas. Além disso, para garantir a factibilidade do problema, existe a necessidade, sem perda de generalidade[4], de que $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$.

Para Sharma (1977), o problema de transporte é provavelmente o mais importante problema de programação linear com características especiais. O autor cita três motivos para isso:

1. Os problemas de transporte formam a primeira classe descoberta de programas lineares a ter matriz totalmente unimodular[5] e pontos extremos inteiros[6], resultando em consideráveis simplificações para o método simplex;
2. Os problemas de transporte lançaram a base para o desenvolvimento teórico e de algoritmos para os problemas de custo mínimo em fluxo em rede;
3. O mais importante, as diversas aplicações em distribuição.

Embora existam problemas de programação linear que possam ser resolvidos de forma exata em tempo polinomial, existem outros tipos de problemas, principalmente os que possuem variáveis inteiras (ou binárias), como o problema das p-medianas, o problema do caixeiro viajante e os problemas com custos fixos (em geral) que não possuem algoritmos que encontrem soluções exatas em tempo polinomial. Desta forma, foram propostas diferentes heurísticas e meta-heurísticas para a resolução destes problemas, que mesmo não existindo a garantia de a solução ser ótima, apresentam boas soluções.

Em Problemas de Transporte, custos fixos podem representar pedágios em rodovias, taxas de pouso em aeroportos, custos de *setup* em sistemas de produção, custos de construção de rodovias, entre outros (PALEKAR, KARWAN, e ZIONTS,

[4] No caso de diferença entre as quantidades totais de oferta e demanda, pode ser criado um ponto fictício (de origem ou destino), conforme procedimento citado na seção 2.3.4 da presente tese.

[5] Matriz totalmente unimodular é aquela em que todas as submatrizes quadradas possuem determinante igual a -1, 0 ou 1.

[6] Por pontos extremos inteiros, entende-se que nos pontos extremos os valores das variáveis são inteiros. Este fato ocorrerá caso os valores de ofertas a_i e demandas b_j sejam inteiros.

1990). Os mesmos autores destacam que, dependendo da aplicação, a importância dos custos fixos no modelo pode variar.

Neste contexto, sendo um caso particular dos problemas de custo fixo[7] e uma generalização do Problema de Transporte, foi proposto por Hirsch e Dantzig (1954), o Problema de Transporte com Custo Fixo (PTCF).

$$\min F(x) = \sum_{i=1}^{m}\sum_{j=1}^{n}\left(c_{ij}x_{ij} + f_{ij}y_{ij}\right) \quad (5)$$

$$\sum_{j=1}^{n} x_{ij} = a_i, i = 1, \dots, m \quad (6)$$

$$\sum_{i=1}^{m} x_{ij} = b_j, j = 1, \dots, n \quad (7)$$

$$x_{ij} \leq m_{ij}y_{ij}, i = 1, \dots, m; j = 1, \dots, n \quad (8)$$

$$y_{ij} \in \{0,1\} \quad (9)$$

$$x_{ij} \geq 0, i = 1, \dots, m; j = 1, \dots, n \quad (10)$$

A função objetivo (5) diferencia-se da função objetivo (1) pela inclusão dos custos fixos, sendo agora necessária a minimização da soma dos custos variáveis com os custos fixos. Os conjuntos de restrições (6), (7) e (10) são os mesmos que os (2), (3) e (4), respectivamente. No conjunto de restrições (8), $m_{ij} = min\{a_i, b_j\}$ e ele implica que somente poderá ocorrer transporte da origem i para o destino j caso a ligação de i para j seja ativada. O conjunto (9) indica que as variáveis y_{ij} são binárias, sendo que $y_{ij} = 1$ implica que existe a ligação de i para j e $y_{ij} = 0$ que não existe.

Os conjuntos (5)-(10) forçam que $y_{ij} = 1$ se e somente se $x_{ij} > 0$ e $y_{ij} = 0$ se e somente se $x_{ij} = 0$. Se $y_{ij} = 0$, o conjunto de restrições (8) faz com que x_{ij} seja 0. Por outro lado, se $x_{ij} = 0$, o conjunto de restrições (8) permite que y_{ij} seja 0 e como a função objetivo (5) é de minimização, a variável y_{ij} assumirá[8] o valor 0.

[7] Para informações sobre problemas gerais de custo fixo, recomenda-se a leitura de Wright e Lanzenauer (1989).

[8] Partindo do pressuposto que o custo fixo f_{ij} é maior do que zero.

Foi mostrado por Hirsch e Dantzig (1968) que o PTCF é um problema do tipo NP-Hard e, de forma complementar, Klose (2008) mostrou ainda que mesmo com somente um ponto de destino, o problema já é NP-hard.

Dentre aplicações diferentes para o problema de transporte com custo fixo, Stroup (1967) apresentou um trabalho sobre alocação de veículos lançadores para missões espaciais, Dutton *et al*. (1974) realizaram uma aplicação para definir a localização de usinas nucleares, Walker (1976) aplicou o modelo para gestão de resíduos sólidos, Hultberg e Cardoso (1997) utilizaram para alocação de professores, entre outras. Ji e Chu (2002) destacam que, entre outras aplicações, o Problema de Transporte pode ser utilizado para resolver problemas de controle de estoque, sequenciamento de horário de funcionários e designação de pessoal.

Para resolução do PTCF, Balinski (1961), Kuhn e Baumol (1962), Cooper e Drebes (1967), Cooper e Olson (1968), Denzler (1969), Robers e Cooper (1969), Steinberg (1970), Walker (1976), Sun *et al*. (1998) e Glover, Amini e Kochenberger (2005) propuseram métodos heurísticos. Além disso, Murty (1968), Steinberg (1970), Gray (1971), Tompkins (1971) e Frank (1972) propuseram métodos exatos.

A motivação inicial no presente ttrabalho foi o PTCF, onde foi observado que os métodos com os melhores resultados na literatura possuem a mesma estrutura de mudança de base que o PT, de forma que os processos existentes para resolução do PT são executados muito mais vezes no PTCF. Diante disso estudou-se com maior profundidade o PT, devido às semelhanças nos conjuntos de restrições e propriedades dos problemas.

Durante o desenvolvimento do trabalho e os resultados obtidos para o PT, decidiu-se por delimitar a tese para o problema sem custo fixo. Como conceitos aqui utilizados podem ser expandidos para trabalhos sobre o PTCF, no apêndice é realizada uma explicação mais detalhada sobre este problema e também são apresentadas nas sugestões para trabalhos futuros como adaptar para o PTCF a estrutura aqui utilizada para o PT.

1.1 OBJETIVO GERAL

O objetivo geral da presente tese é modificar o método tradicional de resolução do Problema de Transporte de modo a obter uma implementação de baixo tempo computacional para o processo de melhoria de solução no Problema de Transporte.

1.2 OBJETIVOS ESPECÍFICOS

A fim de atingir o objetivo geral, os objetivos específicos são:

- Estudar as características do Problema de Transporte
- Estudar diferentes estruturas de dados para representação de soluções básicas factíveis do Problema de Transporte
- Demonstrar, sem a utilização das variáveis duais, que o cálculo dos novos valores de custos atualizados, após a troca de base, pode ser realizado somente utilizando os custos atualizados da iteração anterior.
- Implementar computacionalmente as rotinas desenvolvidas
- Realizar testes computacionais nas rotinas implementadas a fim de identificar onde ocorre o maior gasto de tempo e propor novos métodos para resolução em menor tempo computacional
- Propor um novo procedimento para a escolha da variável não básica a tornar-se básica a cada iteração
- Comparar os tempos computacionais obtidos para as diferentes rotinas e metodologias implementadas

1.3 RELEVÂNCIA E IMPORTÂNCIA

O Problema de Transporte possui algoritmo específico conhecido para a obtenção da solução exata desde a década de 1950, assim o desafio não se constitui em conseguir resolver este problema, mas sim em conseguir resolvê-lo de forma mais rápida.

Embora fosse possível a resolução para o PT por qualquer método de resolução de Programação Linear, métodos específicos que utilizem vantagens das características dos problemas estudados têm importância no desenvolvimento de algoritmos e podem gerar tempos melhores de resolução.

O presente trabalho apresenta modificações estruturais e teóricas no método tradicional de resolução do Problema de Transporte, gerando economia de tempo de resolução quando comparado ao método tradicionalmente conhecido.

Além disto, assim como destaca Silva (2012), o Problema de Transporte, como outros na Pesquisa Operacional, é utilizado como subproblema para a resolução de outros, como o caso em que meta-heurísticas precisem calcular o valor da função objetivo associada ao Problema de Transporte diversas vezes. Assim, uma metodologia que resolva o PT em menor tempo computacional, além de permitir melhora no tempo de resolução para os problemas já conhecidos também pode viabilizar sua utilização para problemas de dimensões maiores.

1.4 ESTRUTURA DO TRABALHO

A presente tese está organizada em seis capítulos. No primeiro introduz-se o tema do trabalho, assim como seus objetivos.

No segundo capítulo, faz-se uma revisão de literatura, na qual explicam-se os principais conceitos de grafos para o desenvolvimento da tese e características do Problema de Transporte.

No capítulo três, é demonstrado, sem a utilização das variáveis duais, quais são os valores de custos atualizados que precisam ser recalculados e que estes

podem ser obtidos utilizando somente os valores dos custos atualizados da iteração anterior.

No quarto capítulo, apresenta-se a implementação em árvore que foi realizada, sendo descritas as estruturas de dados e rotinas utilizadas para a obtenção dos resultados apresentados no capítulo 5. Além disso, é apresentada uma nova proposta de procedimento para a escolha da variável a entrar na base.

No capítulo 5 são apresentados e comparados resultados para diferentes estruturas e procedimentos implementados.

Por fim, no sexto capítulo são realizadas as considerações finais e sugestões para trabalhos futuros.

2 REVISÃO DE LITERATURA

No presente capítulo são apresentados os conceitos já publicados na literatura que servem de base para a elaboração da presente tese. Inicialmente são descritos conceitos de grafos e árvores, a seguir são apresentadas as características especiais do Problema de Transporte e, por fim, realizada uma análise envolvendo a relação de árvores geradoras em grafos bipartidos com o Problema de Transporte.

2.1 CONCEITOS DE GRAFOS

Segundo O'Connor (2002), as definições de raiz, pai, filho, folha, vértice interno, caminho, comprimento de caminho, ancestral, descendente, altura e profundidade são usuais em ciência da computação, mas nem sempre utilizados em teoria dos grafos. Os conceitos e notações apresentados a seguir foram baseados nos trabalhos de Christofides (1975), Gibbons (1985), Chartrand e Oellermann (1993), Carvalho (2002), O'Connor (2002), Sacomoto (2011) e Cormen *et al.* (2012).

Ressalta-se que na literatura não existe uma notação padrão para cada um dos conceitos apresentados, ocorrendo assim variações de um trabalho para outro. Desta forma, serão definidas na presente seção as notações de forma que sejam a mais conveniente para a utilização na presente tese.

Um grafo não orientado G é um par (V, E) onde V é um conjunto finito no qual os elementos são denominados vértices e $E \subseteq \mathrm{V} \times \mathrm{V}$ é um conjunto de pares denominados arestas. Neste caso se uma aresta (u, v) pertence ao conjunto E, então existe ligação entre os vértices u e v. Um exemplo de grafo não orientado sendo $V = \{1, 2, 3, 4, 5\}$ e $E = \{(1,2), (1,4), (2,3), (2,4), (3,1), (3,5), (4,5)\}$, é apresentado na figura 1(a).

Para o caso de existência de orientação, tem-se um grafo orientado D, denominado dígrafo, no qual é dito que um arco (u, v) tem origem em u e destino em

v. Por exemplo, seja $V = \{1, 2, 3, 4, 5\}$ e $E = \{(1,2), (1,4), (2,3), (2,4), (3,1), (3,5), (4,5)\}$, o dígrafo $D = (V, E)$ pode ser representado graficamente conforme figura 1(b).

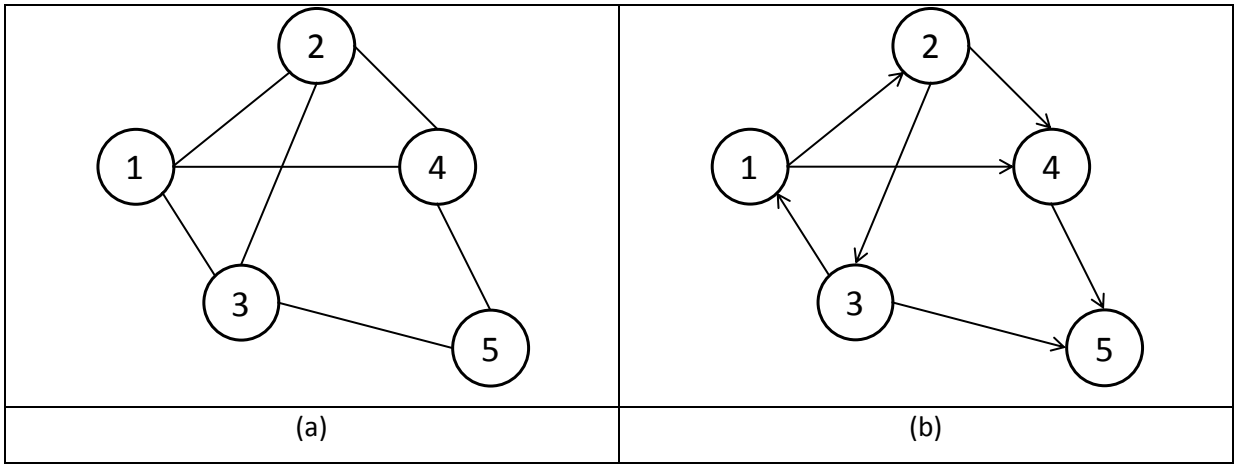


FIGURA 1 – EXEMPLO DE GRAFO E DÍGRAFO
FONTE: O Autor (2014)

Um caminho P de comprimento k de um vértice s para um vértice t no dígrafo $D = (V, E)$ é uma sequência de vértices distintos $\langle y_0, y_1, \dots, y_k \rangle$ tais que $y_0 = s$, $y_k = t$ e $(y_{i-1}, y_i) \in E$ para $i \in \{1, \dots, k\}$. Um caminho Ω também pode ser visto como uma sequência de arcos $(e_1, e_2, \dots, e_k)$, tais que $e_i = (y_{i-1}, y_i)$ para $i \in \{1, \dots, k\}$, ou seja, o vértice de destino do arco e_i é o vértice de origem do arco e_{i+1} e assim todos os arcos estão direcionados para o vértice y_k. Desta forma, o comprimento de um caminho é definido pelo número de arcos contidos nele. Um conceito análogo ao de caminho é o de cadeia, onde a diferença é que os arcos não precisam estar necessariamente todos direcionados para o vértice y_k.

Utilizando o mesmo grafo orientado apresentado na figura 1(b) como exemplo, tem-se que um caminho de comprimento $k = 2$ entre os vértices 1 e 5 é $(1, 4, 5)$. Como $(1, 2, 4, 5)$ também é um caminho entre 1 e 5, tem-se que o caminho entre dois vértices num grafo pode não ser único. Um exemplo de cadeia no dígrafo da figura 1 (b) entre os vértices 4 e 1 seria $(4, 5, 3, 1)$. Outro exemplo para uma cadeia de comprimento 2 entre os vértices 1 e 5 é $(1, 3, 5)$.

Para o grafo não orientado apresentado na figura 1(a), tem-se, por exemplo, uma cadeia $(1, 3, 5)$ de comprimento 2 entre os vértives 1 e 5.

Um subcaminho do caminho $P = \langle v_0, v_1, \dots, v_k \rangle$ é uma sequência contígua de seus vértices. Isto é, para qualquer $0 \leq i \leq j \leq k$, a subsequência de vértices $\langle v_i, \dots, v_j \rangle$ é um subcaminho de P.

Seja $P = \langle x_1, x_2, \dots, x_{r-1}, x_r \rangle$, então denotam-se os subcaminhos de P $\langle x_1, \dots, x_i \rangle$, $\langle x_i, \dots, x_r \rangle$ e $\langle x_i, \dots, x_j \rangle$ por Px_i, x_iP e x_iPx_j, respectivamente. Desta forma, a união de dois caminhos $P_1 = \langle x_1, \dots, x_i \rangle$ e $P_2 = \langle x_i, \dots, x_r \rangle$ resulta no caminho $P = \langle x_1, \dots, x_r \rangle$ e pode ser denotada por $P = P_1 x_i \cup x_i P_2$ ou $P = P_1 x_i P_2$ ou $P = \langle x_1, P_1, x_i, P_2, x_r \rangle$.

Novamente utilizando o grafo representado na figura 1 como exemplo, tem-se que se $P_1 = \langle 1, 2, 4 \rangle$ e $P_2 = \langle 4, 5 \rangle$ então $P = P_1 4 \cup 4 P_2 = P_1 4 P_2 = \langle 1, 2, 4, 5 \rangle$ representa o caminho do vértice 1 ao vértice 5. Analogamente, se $\Omega_1 = \{(1,2), (2,4)\}$ e $\Omega_2 = \{(4,5)\}$ então $\Omega = \Omega_1 4 \Omega_2$.

Um grafo G' é um subgrafo de um grafo G se $V(G') \subseteq V(G)$ e $E(G') \subseteq E(G)$, satisfazendo que se $(i, j) \in V(G')$ então $i, j \in G'$. Para o caso do grafo representado na figura 1, poder-se-ia ter $V(G') = V' = \{1, 2, 4\}$ e $E(G') = E' = \{(1, 2), (1, 4)\}$.

A união de dois grafos $G_1 = (V_1, E_1)$ e $G_2 = (V_2, E_2)$, denotada por $G_1 \cup G_2$, resulta num grafo $G = (V, E)$ tal que $V = V_1 \cup V_2$ e $E = E_1 \cup E_2$. A adição de um vértice v a um grafo G pode ser vista como um caso particular da união entre G e um grafo trivial contendo v, sendo denotada por $G + v$. Analogamente, denota-se por $G + e$ a adição de uma aresta $e = (u, v)$ a G.

A retirada de um conjunto de vértices $X \subseteq V$ é denotada por $G - X = G[V \backslash X]$. Já as retiradas de um vértice $v \in V$ e uma aresta $e \in E$ são denotadas, respectivamente, por $G - v$ e $G - e$.

Um ciclo num grafo é um caminho $\langle y_o, y_1, y_2, \dots, y_k \rangle, y_i \in V$ tal que $y_k = y_0$ e $y_1, y_2, \dots, y_{k-1}$ são distintos dois a dois. Por exemplo, no grafo da figura 1 (a), o caminho $\langle 2,4,5,3,2 \rangle$ é um ciclo. Quando não existem ciclos num grafo, ele é dito acíclico.

Um grafo é dito conexo quando dados dois vértices quaisquer, y_i e y_j, existe uma cadeia entre y_i e y_j. Caso contrário, o grafo é dito não conexo. O grafo da figura 2 (c) é um exemplo de grafo conexo enquanto os grafos das figuras 2 (a) e (b) são exemplos de grafos não conexos.

Um grafo acíclico conexo não dirigido é chamado de árvore livre, ou simplesmente árvore. Caso o grafo seja acíclico e não dirigido, ele será um conjunto

de árvores e denominado floresta. Vale ressaltar que uma floresta pode ser transformada em árvore pela inclusão de arestas que não formem ciclo no grafo. Por outro lado, a retirada de uma aresta de uma árvore transforma o grafo numa floresta. Considerando $G = (V, E)$ um grafo não orientado, é equivalente dizer[9]:

- G é uma árvore livre
- Um par de vértice qualquer (v, w) de G está conectado por apenas um caminho
- G é conexo, mas na remoção de uma aresta qualquer o grafo resultante é desconexo
- G é conexo e $|E| = |V| - 1$
- G é acíclico e $|E| = |V| - 1$
- G é acíclico, mas na adição de qualquer aresta o grafo resultante contém um ciclo.

Nas figuras 2(a), 2(b) e 2(c) são apresentadas representações gráficas de florestas e árvore, respectivamente. Caso na floresta representada na figura 2(a) seja adicionado o arco $(1,7)$, tem-se a floresta da figura 2(b). Ainda, se nesta floresta for adicionado o arco $(4,8)$, tem-se a árvore representada na figura 2(c).

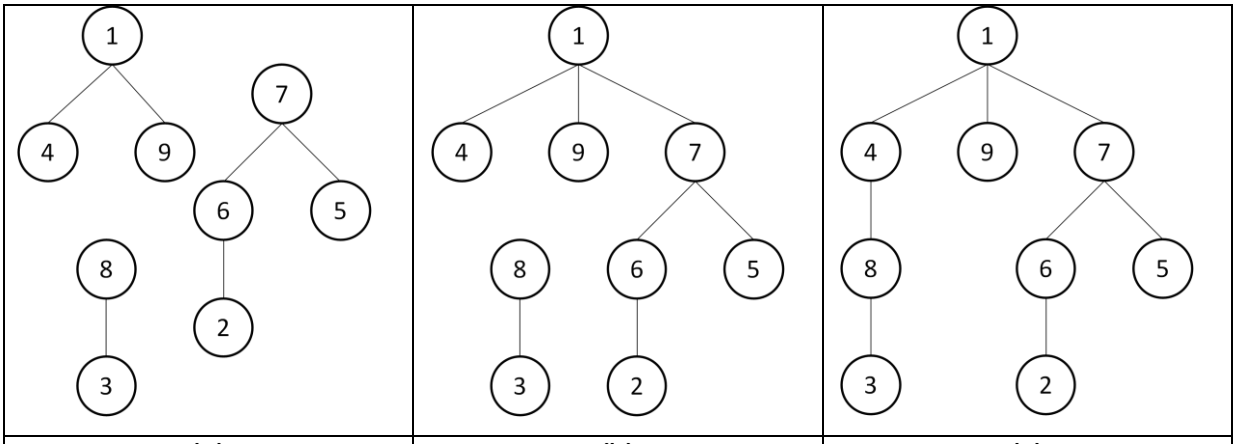


FIGURA 2 – EXEMPLOS DE FLORESTAS E ÁRVORE
FONTE: O Autor (2014)

[9] As demonstrações destas equivalências podem ser consultadas em Cormen *et al*. (2012)

Uma árvore enraizada é um grafo $T = (V, E)$ com um vértice $r \in V$ chamado raiz, tal que para todo vértice $v \in V$, existe um único caminho de r para v. A profundidade de $v \in V$, denotada aqui por $d(v)$, é o comprimento deste caminho e a altura de T é a maior das profundidades.

O vértice u é dito ancestral de v se u pertence ao caminho da raiz até v. Este ancestral é próprio se $u \neq v$. O conjunto dos vértices ancestrais a $v \in V$ será denotado no presente trabalho por $\mathcal{A}(v)$. O menor ancestral comum entre $u, v \in V$, denotado aqui por $\xi(u, v)$, é o ancestral comum entre u e v que possui a maior profundidade.

Se u é um vértice ancestral de v, então v é dito um vértice descendente de u. Caso $u \neq v$, tem-se que v é um descendente próprio de u. Desta forma, a subárvore enraizada em u é a árvore induzida por descendentes de u e com raiz em u. Exemplos de árvores enraizadas são apresentadas na figura 3.

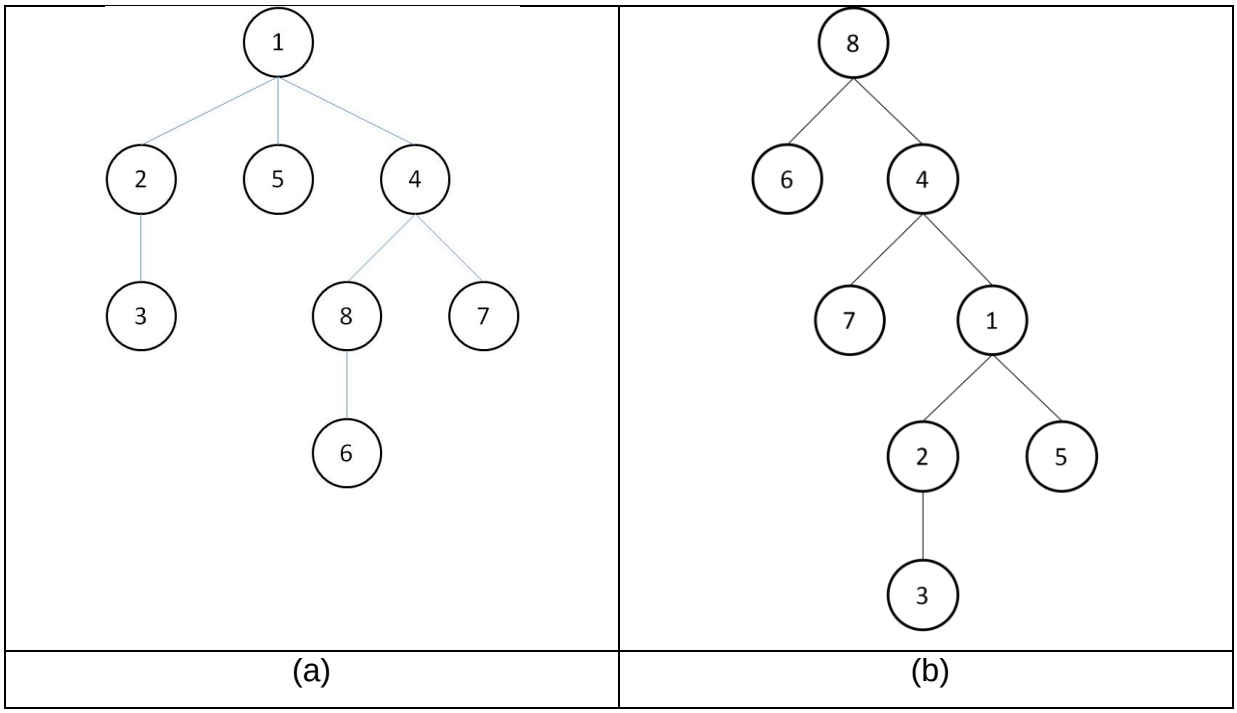


FIGURA 3 – EXEMPLOS DE ÁRVORES ENRAIZADAS
FONTE: O Autor (2014)

A árvore apresentada na figura 3 (a) possui o mesmo conjunto V de vértices e E de arestas que a árvore da figura 3 (b). A diferença entre elas é que uma está enraizada no vértice 1 e a outra no vértice 8. Ou seja, o vértice no qual uma árvore está enraizada altera a sua representação gráfica, assim como o conjunto dos

ancestrais e, consequentemente, a profundidade dos vértices. Entretanto, a cadeia (ou caminho) entre dois vértices $s, t \in V$ permanece o mesmo, independentemente do vértice no qual a árvore está enraizada.

Além disso, uma árvore é dita árvore geradora quando existe ligação entre dois vértices quaisquer pertencentes ao grafo.

Outros conceitos de árvore relevantes para o trabalho são os de pai e filho. Uma forma de defini-los é considerar (u, v) como sendo a última aresta do caminho simples entre a raiz r e o vértice v, então o vértice u é dito pai do vértice v e v é dito filho do vértice u, ou simplesmente u é pai de v e v é filho de u, respectivamente. Desta forma, cada vértice $v \in V - r$ possui um vértice pai, denotado por $p(v)$, e cada vértice $v \in V$ possui um conjunto de filhos, denotado por $b(v)$. Dado um vértice $v \in V$, caso $b(v) = \{\ \}$, v é denominado folha ou vértice externo. Já os vértices que não são folhas, são denominados vértices internos.

De forma semelhante ao conceito de pai, pode ser definido o conceito de avô. Uma forma de defini-los é considerar (s, u) como sendo a penúltima[10] aresta do caminho simples entre a raiz r e o vértice v, então o vértice s é dito avô do vértice v. De forma análoga, poderiam ainda ser definidos os conceitos de bisavô, trisavô, etc.

Quando tem-se um grafo completo com m vértices, o número de árvores geradoras possíveis é m^{m-2}. Já para o caso de grafos bipartidos completos no qual um conjunto possui m vértices e outro n vértices, o número de árvores geradoras é $m^{n-1}n^{m-1}$.

2.2 COMPLEMENTOS SOBRE ÁRVORES GERADORAS

Dos conceitos de grafos, o que será mais utilizado na presente tese é o de árvores geradoras e, por isso, são explicados aqui alguns conceitos complementares sobre elas.

[10] Partindo do pressuposto que o caminho em análise tem comprimento pelo menos 2.

2.2.1 Distância entre árvores geradoras

Segundo Thulasiraman (1992), a distância entre duas árvores geradoras é definida como sendo o número de arestas que estão presentes em uma, mas não estão presentes na outra. O autor denota a distância entre as árvores T_1 e T_2 por

$$d(T_1, T_2) = |T_1 - T_2| = |T_2 - T_1| \quad (11)$$

Considerando que $E(T)$ é o conjunto de vértices da árvore T, Berzukov, Kaderali e Poguntke (1966) definiram a distância entre duas árvores T_1 e T_2 geradoras de G como sendo

$$D(T_1, T_2) = \frac{1}{2}\left|\left(E(T_1) \cup E(T_2)\right) \backslash \left(E(T_1) \cap E(T_2)\right)\right| \quad (12)$$

Ou seja, para os autores a distância é igual a metade da distância simétrica entre $E(T_1)$ e $E(T_2)$. As definições (11) e (12) de distância entre árvores geradoras são equivalentes.

Dado um grafo G, pode-se definir um grafo-árvore denotado por $\mathrm{T(G)}$. Os vértices de $\mathrm{T(G)}$ correspondem a árvores geradoras de G e dois vértices de $\mathrm{T(G)}$ são adjacentes se a distância entre as árvores geradoras correspondentes for 1 (BERZUKOV, KADERALI e POGUNTKE, 1966).

Considerando um grafo bipartido completo (figura 4) com $m = 3$ e $n = 2$, denominado $K_{3,2}$, tem-se $3^{2-1}.2^{3-1} = 12$ árvores geradoras possíveis, as quais são representadas na figura 5.

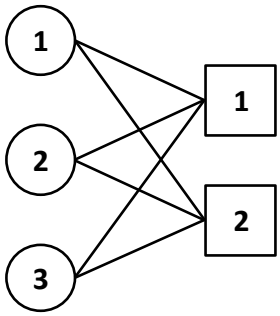


FIGURA 4 – REPRESENTAÇÃO DE UM GRAFO BIPARTIDO $K_{3,2}$

FONTE: O Autor (2014)

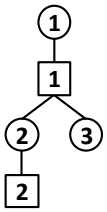
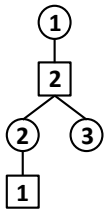
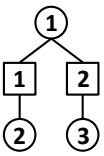
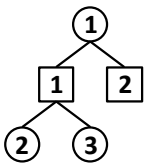
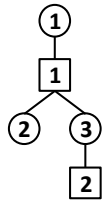
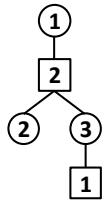
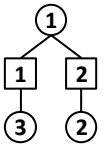
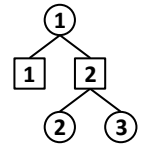
					
(a) A_1	(b) A_2	(c) A_3	(d) A_4	(e) A_5	(f) A_6
					
(g) A_7	(h) A_8	(i) A_9	(j) A_{10}	(k) A_{11}	(l) A_{12}

FIGURA 5 – EXEMPLO DE PRESENTAÇÃO DAS ÁRVORES GERADORAS PARA UM GRAFO BIPARTIDO $K_{3,2}$
FONTE: O Autor (2014)

A matriz de adjacência para as árvores apresentada na figura 5, é representada na figura 6.

	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}
A_1	0	0	1	1	1	0	1	0	1	0	0	1
A_2	0	0	1	1	0	1	0	1	0	1	1	0
A_3	1	1	0	0	0	1	1	0	1	0	1	0
A_4	1	1	0	0	1	0	0	1	0	1	0	1
A_5	1	0	0	1	0	1	0	1	1	0	0	1
A_6	0	1	1	0	1	0	0	1	1	0	1	0
A_7	1	0	1	0	0	0	0	0	1	1	1	1
A_8	0	1	0	1	1	1	0	0	1	1	0	0
A_9	1	0	1	0	1	1	1	1	0	0	0	0
A_{10}	0	1	0	1	0	0	1	1	0	0	1	1
A_{11}	0	1	1	0	0	1	1	0	0	1	0	1
A_{12}	1	0	0	1	1	0	1	0	0	1	1	0

FIGURA 6 – MATRIZ DE ADJACÊNCIA PARA O GRAFO DE ÁRVORES DA FIGURA 5
FONTE: O Autor (2014)

Com base na matriz de adjacência é possível identificar caminhos entre árvores geradoras. Por exemplo, a árvore A_3 não é adjacente a árvore A_{10}, mas pelo caminho $A_3 - A_9 - A_2 - A_{10}$ é possível obter a árvore A_{10} a partir da árvore A_3, sendo necessários neste caso três inclusões e exclusões sucessivas de arestas. Um outro caminho entre as mesmas árvores seria $A_3 - A_2 - A_{10}$, sendo necessárias duas inclusões e exclusões sucessivas de arestas. O menor caminho entre duas árvores poderia ainda ser encontrado por meio da utilização de um algoritmo de caminho mínimo, tendo como base a matriz de adjacência da figura 6.

2.2.2 Árvore geradora central

Deo (1966) introduziu o conceito de árvore geradora central em um grafo G como sendo aquela que a maior distância para todas as árvores geradoras em G é mínima. Desta forma, a árvore geradora T_0 é árvore geradora central de G se

$$\max_i d(T_0, T_i) \leq \max_i d(T, T_i), \forall\, T \in G \qquad (13)$$

Os problemas relacionados a encontrar a árvore geradora central foram estudados intensivamente nas décadas de 1960 e 1980 por Malik (1968), Shinoda e Kawamoto (1980), Shinoda e Kawamoto (1981) e Shinoda e Saishu (1980). Um algoritmo para resolver o problema em tempo polinomial foi proposto por Amoia (1971), mas Kaderali (1973) encontrou um contra exemplo para o método. Anteriormente, Berzukov, Kaderali e Poguntke (1966) já havia demonstrado que o problema é NP-completo.

2.2.3 Máxima distância entre árvores geradoras

O problema de máxima distância entre árvores geradoras consiste em encontrar um par de árvores geradoras T_1 e T_2 tal que a distância entre elas é máxima no grafo de árvores geradoras $T(G)$.

Kishi e Kajitani (1969) demonstraram, baseados no trabalho de Deo (1966), condições para um par de árvores ser de máxima distância.

2.2.4 Estrutura de dados para armazenamento da árvore

A eficiência de um código computacional na implementação de um algoritmo depende, entre outras coisas, da estrutura de dados utilizada (FOX, 1978). Diante disso, são aqui apresentadas algumas das estruturas presentes na literatura para armazenamento de árvores geradoras.

Codificação de Prüfer para árvores geradoras

O código de Prüfer (ou número de Prüfer) representa uma árvore geradora de um grafo de n vértices numa sequência de $n-2$ valores. Quando o grafo é completo, existe uma correspondência única entre cada árvore geradora e cada código de Prüfer (CHEN e WANG, 2000).

Algoritmos diversos foram propostos para a codificação de uma árvore geradora num código de Prüfer e também para o processo inverso de decodificação. Os algoritmos aqui apresentados são baseados em Harris, Hisrt e Mossinghoff (2008).

Passo 1. Defina $i = 0$ e $T_0 = T$
Passo 2. Encontre a folha de T_i com o menor rótulo e a chame de v
Passo 3. Registre na sequência o rótulo do vizinho de v
Passo 4. Remova v de T_i criando uma nova árvore T_{i+1}
Passo 5. Se $T_{i+1} = K_2$, fim. Caso Contrário, faça $i = i + 1$ e retorne ao passo 2.

QUADRO 1 – ALGORITMO PARA REPRESENTAR UMA ÁRVORE GERADORA COMO UM CÓDIGO DE PRÜFER

FONTE: Adaptado de Harris, Hirst e Mossinghoff (2008)

Para exemplo didático do algoritmo apresentado no quadro 1 será utilizada a árvore geradora representada na figura 7.

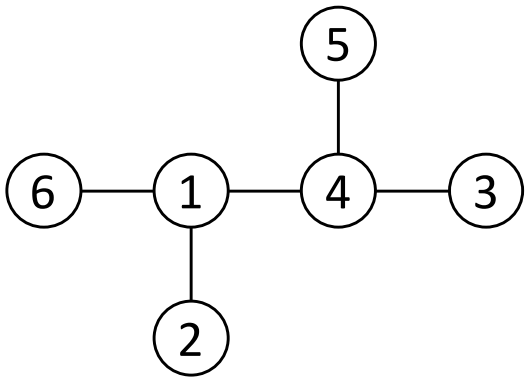
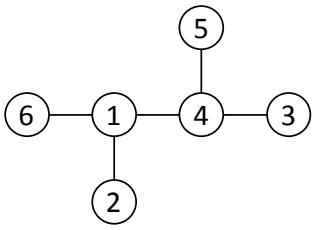
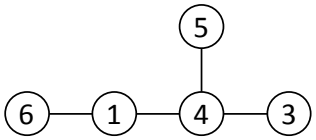
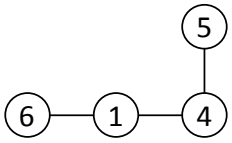
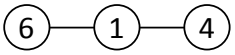
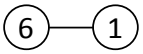


FIGURA 7 – EXEMPLO DE ÁRVORE GERADORA A SER UTILIZADA PARA A CODIFICAÇÃO DE PRÜFER

FONTE: O Autor (2014)

A primeira árvore geradora T_0 (figura 7) é a própria árvore geradora original. As folhas da árvore T_0 são os vértices com índices pertencentes ao conjunto $\{2, 3, 5, 6\}$, sendo 2 o menor destes valores. O vértice v_2 está conectado ao vértice v_1 e, desta forma, o valor 1 passa a fazer parte da sequência P.

O vértice v_2 é então removido da árvore geradora T_0, obtendo-se a árvore T_1. Nesta nova árvore geradora as folhas são os vértices com índices pertencentes ao conjunto $\{3, 5, 6\}$, sendo 3 o menor destes valores. O vértice v_3 está conectado ao vértice v_4 e, desta forma, o valor 4 passa a fazer parte da sequência P. O processo é repetido (quadro 2) até que existam apenas dois vértices na árvore geradora.

i	T_i	$Folhas(T_i)$	P
0		$\{2, 3, 5, 6\}$	$[1]$
1		$\{3, 5, 6\}$	$[1\ 4]$
2		$\{5, 6\}$	$[1\ 4\ 4]$
3		$\{4, 6\}$	$[1\ 4\ 4\ 1]$
4		Fim. $P = [1\ 4\ 4\ 1]$	

QUADRO 2 – PASSO A PASSO DE UM EXEMPLO PARA CODIFICAÇÃO DE PRÜFER DE UMA ÁRVORE GERADORA

FONTE: O Autor (2014)

O algoritmo apresentado por Harris, Hisrt e Mossinghoff (2008) para obter a representação gráfica de uma árvore geradora a partir de um código de Prüfer é descrito no quadro 3.

Dada uma sequência σ de k valores pertencentes ao conjunto $\{1, 2, \dots, k+2\}$
Passo 1. Desenhe $k+2$ vértices rotulados $v_1, v_2, \dots, v_{k+2}$ e defina $S = \{1, 2, \dots, k+2\}$
Passo 2. Defina $i = 0$, $\sigma_0 = \sigma$ e $S_0 = S$
Passo 3. Defina j como sendo o menor valor em S_i que não aparece na sequência σ_i.
Passo 4. Crie uma ligação entre o vértice v_j e o vértice cujo índice aparece primeiro na sequência σ_i.
Passo 5. Remova o primeiro número da sequência σ_i para criar a sequência σ_{i+1}. Remova o elemento j do conjunto S para criar um novo conjunto S_{i+1}.
Passo 6. Se a sequência σ_{i+1} é vazia, crie uma ligação entre dois vértices cujos índices estão em S_{i+1} e fim. Caso contrário, faça $i = i + 1$ e retorne ao passo 3.

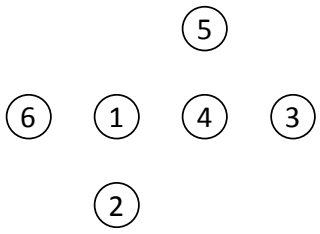
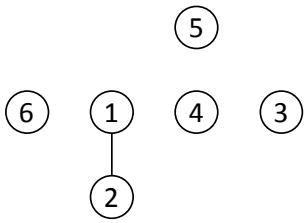
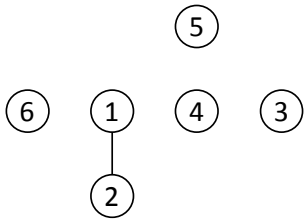
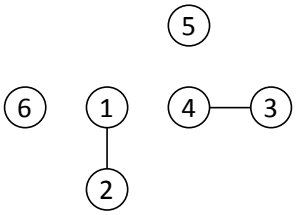
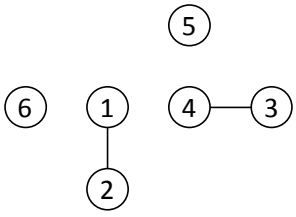
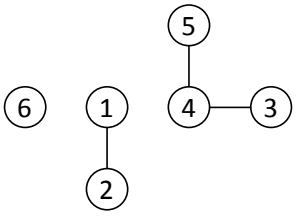
QUADRO 3 – ALGORITMO PARA REPRESENTAÇÃO GRÁFICA DE UMA ÁRVORE GERADORA A PARTIR DE UM CÓDIGO DE PRÜFER.
FONTE: Adaptado de Harris, Hisrt e Mossinghoff (2008)

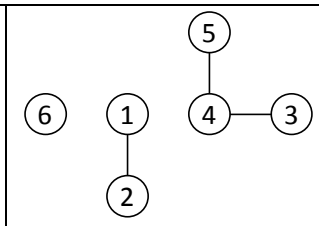
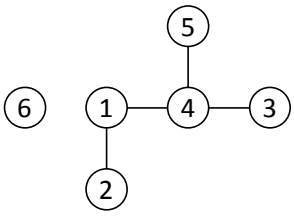
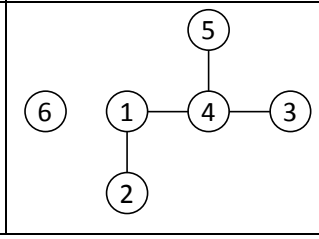
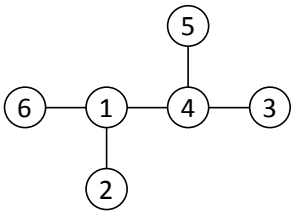
Como exemplo didático para o algoritmo apresentado no quadro 3, será utilizada a sequência $P = [1\ 4\ 4\ 1]$.

A primeira sequência $\sigma_0 = [1\ 4\ 4\ 1]$ é o código de Prüfer a ser decodificado para determinar a representação gráfica da árvore geradora e $S_0 = \{1, 2, 3, 4, 5, 6\}$ é o conjunto dos índices dos vértices do grafo. O menor valor j que pertence ao conjunto S_0 e não pertence ao conjunto σ_0 é 2. Além disso, o primeiro elemento da sequência σ_0 é o valor 1. Desta forma, o vértice v_2 é conectado ao vértice v_1.

Os conjuntos σ_1 e S_1, são obtidos, respectivamente, pela exclusão do valor 1 de σ_0 e do valor 2 de S_0. Para esta nova situação, o menor valor pertencente ao conjunto S_1 e não pertencente a σ_1 é 3. Além disso, o primeiro elemento da sequência σ_0 é o valor 4. Desta forma, o vértice v_3 é conectado ao vértice v_4.

O processo continua de forma iterativa até que não existam mais elementos em σ_i e restem dois elementos em S_i. Neste último passo, os dois elementos de S_i são conectados e tem-se como resultado a árvore geradora associada ao código σ. O passo a passo para o exemplo é apresentado no quadro 4.

i		σ_i	S_i	
0		$[1\ 4\ 4\ 1]$	$\{1, 2, 3, 4, 5, 6\}$	
1		$[4\ 4\ 1]$	$\{1, 3, 4, 5, 6\}$	
2		$[4\ 1]$	$\{1, 4, 5, 6\}$	

3		[1]	$\{1,4,6\}$	
4		[ ]	$\{1,6\}$	

QUADRO 4 – PASSO A PASSO DE UM EXEMPLO PARA REPRESENTAÇÃO GRÁFICA DE UMA ÁRVORE GERADORA A PARTIR DE UM CÓDIGO DE PRÜFER.
FONTE: O Autor (2014)

As principais vantagens da codificação de Prüfer são a fácil representação em estrutura de dados, uma vez que um vetor de dimensão menor do que o número de vértices no grafo é suficiente para o registro da árvore geradora e também a utilidade em demonstrações de propriedades de árvores geradoras.

Estrutura descrita por Gabriel Valiente

Uma classe para representação de árvore enraizada é descrita por Valiente (2010), sendo consideradas as seguintes operações:

Pai(v)	Retorna o pai do nó v, ou nulo para o caso de v ser raiz da árvore
É_raiz(v)	Retorna verdadeiro caso o nó v seja raiz da árvore e falso caso contrário
É_folha(v)	Retorna verdadeiro caso o nó v seja folha da árvore e falso caso contrário
Primeiro_filho(v)	Retorna o primeiro filho do nó v, ou nulo para o caso de v ser um nó folha
Ultimo_filho(v)	Retorna o último filho do nó v, ou nulo para o caso de v ser um nó folha
Proximo_irmao(v)	Retorna o próximo irmão do nó v, ou nulo para o caso de v ser raiz ou último filho

Anterior_irmao(v)	Retorna o irmão anterior do nó v, ou nulo para o caso de v ser raiz ou primeiro filho
É_primeiro_filho(v)	Retorna verdadeiro se o nó v é o primeiro filho e falso caso contrário
É_ultimo_filho(v)	Retorna verdadeiro se o nó v é o último filho e falso caso contrário
Número_de_filhos(v)	Retorna o número de filhos do nó v

QUADRO 5 – ESTRUTURA PARA ARMAZENAMENTO DE INFORMAÇÕES DE UMA ÁRVORE GERADORA
FONTE: Adaptado de Valiente (2010)

O mesmo autor ainda apresenta de forma detalhada a implementação da classe, que será omitida no presente trabalho. Um exemplo para as operações descritas no quadro 5 para a árvore da figura 3 (a) é apresentado no quadro 6.

v	1	2	3	4	5	6	7	8
Pai(v)	0	1	2	1	1	8	4	4
É_raiz(v)	Sim	Não	Não	Não	Não	Não	Não	Não
É_folha(v)	Não	Não	Sim	Não	Sim	Sim	Sim	Não
Primeiro_filho(v)	2	3	0	8	0	0	0	6
Ultimo_filho(v)	4	3	0	7	0	0	0	6
Proximo_irmao(v)	0	5	0	0	4	0	0	7
Anterior_irmao(v)	0	0	0	5	2	0	8	0
É_primeiro_filho(v)	Sim	Sim	Sim	Não	Não	Sim	Não	Sim
É_ultimo_filho(v)	Sim	Não	Sim	Sim	Não	Sim	Sim	Não
Número_de_filhos(v)	3	1	0	2	0	0	0	1

QUADRO 6 – EXEMPLO DE REPRESENTAÇÃO DE UMA ÁRVORE NA ESTRUTURA DE Valiente (2010)

FONTE: O Autor (2014)

A estrutura descrita por Valiente (2010) possui informações que podem ser consideradas redundantes. Por exemplo, caso existissem somente as informações de Pai, Primeiro_filho e Proximo_irmao já seria possível realizar a representação da árvore. Alguns autores utilizam apenas estas três informações de cada vértices para armazenamento da árvore, mas as informações adicionais podem ser utilizadas

conforme conveniência na implementação realizada dependo da necessidade de informações no algoritmo programado. Sustarcic (1999), por exemplo, utilizou para cada nó as informações de predecessor (pai), sucessor (filho), irmão mais novo e irmão mais velho.

2.2.5 Troca de raiz de uma árvore geradora

Para explicação do processo de troca de raiz e as consequentes mudanças serão utilizadas as árvores apresentadas na figura 8, sendo que as setas indicam o sentido da raiz para as folhas. O processo aqui explicado é uma adaptação do exposto em Christofides (1975).

A árvore apresentada na figura 8 (a) está enraizada no vértice 1. Para realizar a troca de raiz de um vértice para outro, deve ser observado o caminho entre a raiz atual e a nova raiz. Por exemplo, na figura 8 (b) é destacado o caminho do vértice 1 (raiz atual) para o vértice 7 (nova raiz).

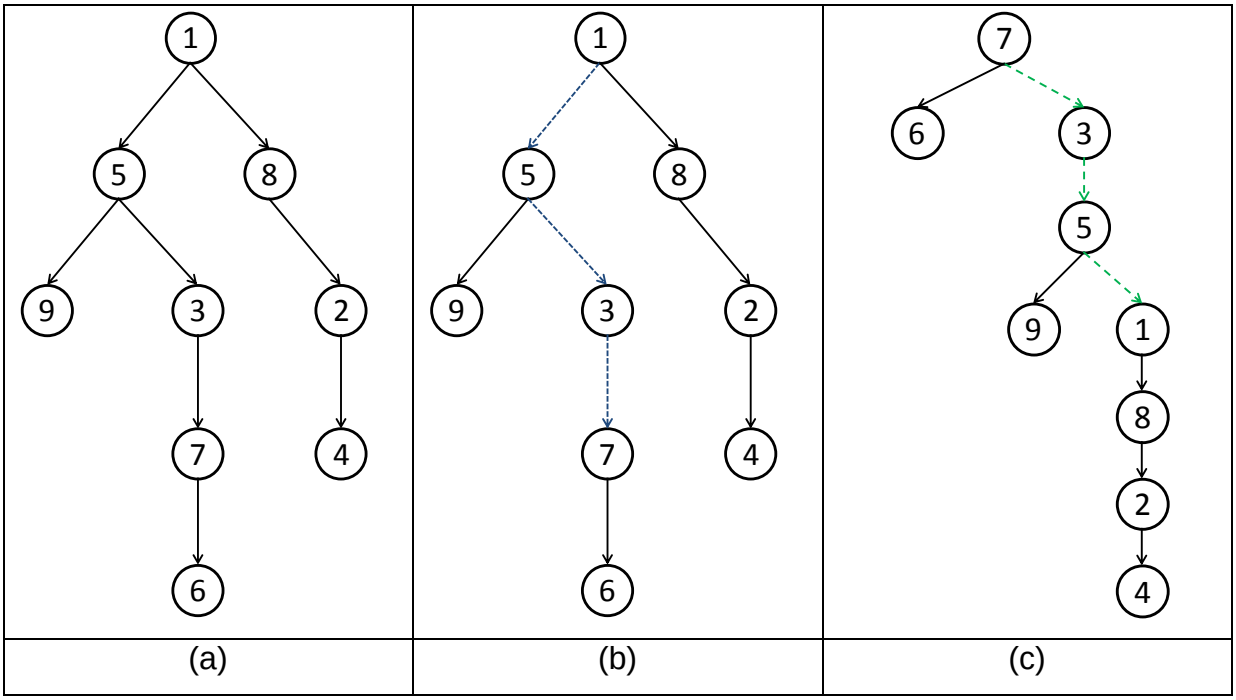


FIGURA 8 – EXEMPLO PARA A EXPLICAÇÃO DE TROCA DE RAIZ EM ÁRVORES GERADORAS
FONTE: O Autor (2014)

Após identificado o caminho entre a raiz atual e a nova raiz, o próximo passo é a inversão dos sentidos dos arcos no caminho entre os dois vértices. Para o exemplo, tem-se que a árvore enraizada no vértice 7 é a apresentada na figura 8 (c).

As setas nas árvores da figura 8 estão no sentido da raiz para as folhas e, consequentemente, no sentido de pai para filho. No quadro 7 são apresentados o pai e os filhos de cada vértice da figura 8 (a).

Vértice	1	2	3	4	5	6	7	8	9
Pai	0	8	5	2	1	7	3	1	5
Filhos	{5, 8}	4	{7}	{}	{3,9}	{}	{6}	{2}	{}

QUADRO 7 – INFORMAÇÕES DE PAI E FILHOS DE CADA VÉRTICE DA ÁRVORE GERADORA DA FIGURA 8 (a)
FONTE: O Autor (2014)

Quando ocorre mudança de raiz numa árvore, os únicos vértices que possuem o pai e o conjunto de filhos alterados são aqueles que pertencem ao caminho da raiz atual para a nova raiz. Para o exemplo, no quadro 8 são apresentados o pai e os filhos de cada vértice da árvore da figura 8 (c).

Vértice	1	2	3	4	5	6	7	8	9
Pai	**5**	8	**7**	2	**3**	7	**0**	1	5
Filhos	**{8}**	4	**{5}**	{}	**{1**,9**}**	{}	**{**6,3**}**	{2}	{}

QUADRO 8– INFORMAÇÕES DE PAI E FILHOS DE CADA VÉRTICE DA ÁRVORE GERADORA DA FIGURA 8 (c)
FONTE: O Autor (2014)

Desta forma, tem-se que procedimento de troca de raiz de árvores não envolve a necessidade de atualização de todos os valores de pai e filhos de cada nó. Além disso, quanto menor for o caminho entre a raiz atual e a nova raiz, menor será o número de atualizações necessárias.

Poderiam ainda ser atualizadas as outras informações, como as apresentadas em Valiente (2010). Porém, as informações de Proximo_Irmao, Anterior_Irmao, Primeiro_Filho e Ultimo_Filho tem como finalidade evitar a dupla representação gráfica de um árvore.

2.3 CARACTERÍSTICA DO PROBLEMA DE TRANSPORTE

Como já citado anteriormente, o PT possui características especiais que podem ser utilizadas para o desenvolvimento de métodos específicos mais eficientes para a obtenção de sua solução ótima. Nesta seção são apresentadas as características que são utilizadas para o desenvolvimento teórico e implementação computacional apresentadas posteriormente na presente tese.

2.3.1 Existência de solução ótima com $m + n - 1$ variáveis básicas

Conforme citado por Murty (1983) e Bazaraa, Jarvis e Sherali (2010), existe exatamente uma restrição redundante de igualdade no conjunto de restrições do PT. Além disso, quando qualquer uma das restrições é retirada, as remanescentes são linearmente independentes.

Desta forma, como o número de restrições é $m + n$, tem-se $m + n - 1$ restrições linearmente independentes, de forma que uma solução básica para o problema, depois de eliminada uma restrição, será constituída de no máximo $m + n - 1$ variáveis com valores diferentes de zero.

Por outro lado, num problema de programação linear, quando existe solução ótima, existe também pelo menos uma solução ótima num ponto extremo da região factível. Como um ponto extremo é uma solução básica, existirá uma solução básica que seja ótima para o problema e, consequentemente, existirá uma solução ótima com $m + n - 1$ variáveis básicas. Isto é, existirá uma solução ótima com no máximo $m + n - 1$ variáveis com valores diferentes de zero.

2.3.2 Solução com valores inteiros e vantagens computacionais

Conforme demonstrado em Murty (1983), as soluções dos problemas primal e dual podem ser obtidas utilizando somente operações de adição e subtração e os valores das variáveis básicas x_{ij} serão da forma $x_{ij} = \sum_{i=1}^{m} \gamma_i a_i + \sum_{j=1}^{n-1} \psi_j b_j$, sendo que $\gamma_i, \psi_j \in \{-1, 0, 1\}$. Logo, se $a, b \in \mathbb{Z}$, as variáveis básicas, por serem adições e subtrações de valores inteiros, também serão valores inteiros, ou seja, $x \in \mathbb{Z}$.

Em linguagens de programação, operações envolvendo variáveis que são do tipo inteiras possuem tempos computacionais de execução menores do que quando as variáveis são do tipo ponto flutuante.

2.3.3 Representação em quadro e árvore para o problema do transporte

Antes de descrever sobre métodos para a resolução do PT, faz-se necessária a definição de estruturas que são convenientes de serem utilizadas para a representação de um PT.

Um Problema de Transporte pode ser apresentado numa estrutura denominada quadro de transporte (figura 9), o qual contém informações sobre os custos associados a cada ligação origem-destino, a identificação das variáveis básicas e os valores atuais de cada variável. (SILVA, 2012)

Destino / Origem	1	2	⋯	n	Oferta
1	c_{11} x_{11}	c_{12} x_{12}	⋯	c_{1n} x_{1n}	a_1
2	c_{21} x_{21}	c_{22} x_{22}	⋯	c_{2n} x_{2n}	a_2
⋮	⋮	⋮	⋱	⋮	⋮
m	c_{m1} x_{m1}	c_{m2} x_{m2}	⋯	c_{mn} x_{mn}	a_m
Demanda	b_1	b_2	⋯	b_n	$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$

FIGURA 9 – REPRESENTAÇÃO GENÉRICA DO QUADRO DE TRANSPORTE
FONTE: O Autor (2014)

Como exemplo de um PT representado em quadro com uma solução básica factível tem-se o quadro da figura 10.

Destino / Origem	1	2	3	4	Oferta
1	5 100	6	8	2	100
2	2 50	5 80	7 70	1	200
3	6	3	3 20	4 30	50
Demanda	150	80	90	30	350

FIGURA 10 – EXEMPLO DE PROBLEMA DE TRANSPORTE REPRESENTADO EM QUADRO
FONTE: O Autor (2014)

Além disso, um PT também pode ser representado por um grafo, conforme descrito em Bazaraa, Jarvis e Sherali (2010) e apresentado detalhadamente em Silva (2012). O grafo que representa um PT possui a característica especial de ser bipartido, ou seja, existem ligações somente no sentido Origem-Destino.

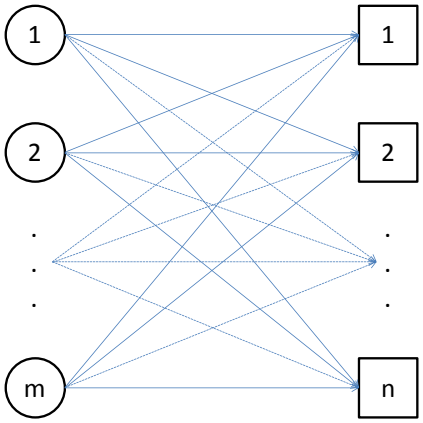


FIGURA 11 – REPRESENTAÇÃO DO PROBLEMA DO TRANSPORTE EM GRAFO
FONTE: O Autor (2014)

Uma solução básica na estrutura em quadro é identificada pela não existência de ciclos no quadro com $m + n - 1$ células alocadas. Já no grafo, a solução básica é representada por uma árvore geradora. A relação entre soluções básicas no problema do transporte e estruturas de árvore foi inicialmente apresentada por Koopmans (1949). A solução básica do quadro da figura 10 é representada em grafo pela imagem da figura 12.

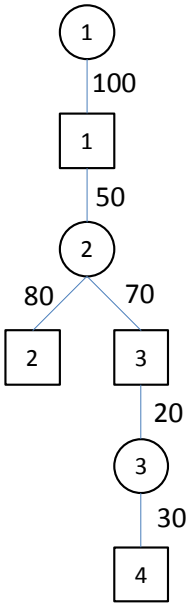


FIGURA 12 – EXEMPLO DE SOLUÇÃO BÁSICA EM GRAFO PARA O PT
FONTE: O Autor (2014)

Para o caso específico do Problema do Transporte na presente tese, as definições e conceitos referentes à arvores serão adaptados de forma conveniente para facilitar a explicação e entendimento do método proposto.

No caso do PT, os vértices representam nós de Origem e Destino. Desta forma, $\mathcal{O}_i$ denota o $i-ésimo$ nó de origem e é representado graficamente por círculo, já $\mathcal{D}_j$ denota o $j-ésimo$ nó de destino e é representado graficamente por quadrado. Os conjuntos $\mathcal{O} = \{\mathcal{O}_1, \dots, \mathcal{O}_m\}$ e $D = \{\mathcal{D}_1, \dots, \mathcal{D}_n\}$ representam os conjuntos de todos os nós de origem e destino, respectivamente. Assim, o conjunto de arcos[11] $E = \mathcal{O}x\mathcal{D}$. Por simplificação de notação, será considerado $\mathcal{K} = \mathcal{O} \cup \mathcal{D}$.

A solução básica factível $\mathcal{B}$ para o PT pode ser representada por meio de uma árvore de transporte enraizada em r, denotada por $\mathcal{T}(\mathcal{B}, r)$. Por simplificação de notação, considera-se que $r = \mathcal{O}_1$ em caso de omissão.

Um PT com sete origens, $\mathcal{O} = \{\mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3, \mathcal{O}_4, \mathcal{O}_5, \mathcal{O}_6, \mathcal{O}_7\}$, e seis destinos, $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_4, \mathcal{D}_5, \mathcal{D}_6\}$, que possui como solução básica $\mathcal{B} = \{(1,4), (1,3), (1,6), (2,2), (3,4), (3,5), (3,2), (4,2), (5,2), (6,6), (6,1), (7,6)\}$ pode ter como árvore associada $\mathcal{T}_{\mathcal{B}\mathcal{O}_1}$, a qual é representada graficamente na figura 13. Ressalta-se que este tipo de representação é baseado no trabalho de Silva (2012) e as direções dos arcos são omitidas por serem sempre na direção de um vértice $s \in \mathcal{O}$, representados por círculos, para um vértice $t \in \mathcal{D}$, representados por quadrados.

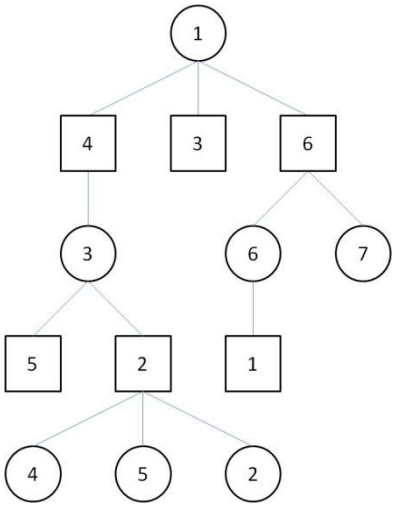


FIGURA 13 – EXEMPLO DE ÁRVORE ENRAIZADA PARA O PROBLEMA DE TRANSPORTE
FONTE: O Autor (2014)

[11] Na presente tese são considerados problemas densos, ou seja, com a existência de todas as possíveis ligações.

2.3.4 Resolução do Problema de Transporte

Quando dado um PT no qual a soma das ofertas é maior que a soma das demandas ($\sum_{i=1}^{m} a_i \geq \sum_{j=1}^{n} b_j$), tem-se um excesso de oferta, já no caso de a soma das demandas ser maior que a soma das ofertas ($\sum_{j=1}^{n} b_j \geq \sum_{i=1}^{m} a_i$), tem-se um excesso de demanda. Os métodos apresentados na literatura partem da hipótese de que a soma das ofertas é igual a soma das demandas. Isto pode ser utilizado sem perda de generalidade, pois no caso do excesso de oferta pode-se inserir um destino (coluna) artificial no problema com a demanda sendo igual ao excesso de oferta ($b_{n+1} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j$), para representar a folga, com custo zero de transporte a partir de cada origem. Analogamente, para o caso de excesso de demanda, insere-se uma origem (linha) artificial com a oferta sendo igual ao excesso de demanda ($a_{m+1} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i$), para representar o excesso de demanda, com custo zero de transporte para cada destino. Este procedimento é amplamente conhecido na literatura, sendo descrito, entre outros, por Bazaraa, Jarvis e Sherali (2010), Arenales *et al*. (2007), Murty (1983) e Silva (2012).

O Problema de Transporte pode ser resolvido por meio do método Simplex tradicional. Entretanto, devido ao fato de o Problema de Transporte também ser representado como um problema de fluxo em rede, ele pode ser resolvido pelo simplex em rede[12]. Ainda, mais especificamente, utilizando a estrutura particular do problema, foi desenvolvido o algoritmo de transporte, o qual tem passos gerais similares a diversos outros métodos de resolução de programação linear e é composto dos seguintes passos:

Passo 1. Obter uma Solução Básica Factível Inicial (SBFI).

Passo 2. Verificar se critério de otimalidade foi atingido. Se foi atingido, fim.

Passo 3. Realizar melhoria da solução. Retornar ao passo 2.

O que caracteriza o algoritmo de transporte não são os três passos gerais, mas sim como eles são executados aproveitando a estrutura do problema.

[12] Para detalhes sobre o simplex em rede, recomenda-se a leitura do capítulo sete de Bazaraa, Jarvi e Sherali (2010)

2.3.5 Obtenção da solução básica factível inicial

Para encontrar a Solução Básica Factível Inicial (SBFI) muitos métodos foram desenvolvidos (SOUZA, 2004), tais como o método do Canto Noroeste, método de Vogel, método do Custo Mínimo e método de Russell, sendo os três primeiros os mais difundidos na literatura.

O método do canto noroeste não considera os custos de transporte, ou seja, são utilizadas somente as informações das ofertas e das demandas (WISTON, 2004). Como vantagem, tem-se a simples implementação computacional e como desvantagem o fato de o valor da função objetivo ter grandes chances de estar distante do ótimo.

Para o caso do método do custo mínimo, conforme descrito por Puccini e Pizzolato (1987), a solução inicial depende dos valores das ofertas, das demandas e dos custos de transporte, visando uma solução inicial mais próxima do ótimo do que a fornecida pelo Canto Noroeste. Entretanto, vale destacar que não existem garantias de que a solução obtida será melhor do que a gerada pelo método do canto noroeste. O método do custo mínimo pode ser classificado como um método guloso[13].

Vogel propôs um método em que a cada escolha de variável básica deve-se calcular para cada linha e para cada coluna a diferença entre os dois menores custos, denominada penalização. Na linha ou coluna onde ocorrer a maior penalização, escolhe-se então a célula que tiver o menor custo. A variável associada a esta célula é a que será escolhida como básica.

No método de Russell, para cada célula subtrai-se do custo o maior custo da linha e o maior custo da coluna. A variável associada à célula que possuir o menor valor é então escolhida para ser básica e o processo é repetido iterativamente.

A ideia de possuir uma solução inicial com valores mais próximos do valor ótimo é a expectativa de que neste caso sejam necessárias menos iterações e, consequentemente, menor tempo computacional para que seja alcançada uma solução ótima para o problema.

[13] Métodos gulosos são aqueles em que as decisões são tomadas com base nas informações disponíveis na iteração corrente, sem considerar as consequências destas decisões nas próximas iterações.

Na literatura, poucos trabalhos existem para comparação destes métodos. Joshi (2013) apresenta um exemplo de problema de transporte de tamanho $3x3$ e compara os valores da função objetivo quando utilizados os métodos do canto noroeste, custo mínimo, Vogel com o da solução exata.

Nos últimos anos, alguns autores propuseram métodos que são apresentados como exatos para a solução do PT, mas não apresentam demonstração alguma de que são de fato exatos. Mesmo não sendo métodos exatos, faz-se importante a implementação computacional dos mesmos para comparação do desempenho como solução inicial, uma vez que o número de iterações para que a solução ótima seja atingida pode ser menor. No presente trabalho não se questiona a qualidade destes métodos, eles são aqui apresentados por serem os que aparecem em buscas sobre trabalhos recentes, publicados em periódicos, que abordam propostas de solução para o Problema de Transporte.

Estes trabalhos são os de Deshmukh (2012), Hakim (2012), Palaniyappa e Vinoba (2013), Aramuthakannan e Kandasamy (2013), Soomro, Tularam e Bhayo (2014) e Somani e Somani (2014).

2.3.6 Variáveis básicas degeneradas no Problema de Transporte

Em modelos de programação linear, variáveis básicas degeneradas são variáveis básicas que possuem valores que poderiam ser assumidos por variáveis não básicas (BAZARAA, JARVIS e SHERALI, 2010). Para o caso específico do PT, as únicas variáveis que podem assumir valores diferentes de zero são as básicas. Entretanto, é possível a existência de variáveis básicas que possuem valor zero e, neste caso, estas são chamadas de variáveis básicas degeneradas.

Um problema que pode surgir para determinados métodos de solução básica inicial é o de como completar a base com variáveis degeneradas. Babu *et al*. (2014) citam que quando o método de Vogel é utilizado para a obtenção de uma solução inicial, pode acontecer de as ofertas e as demandas se esgotarem ao mesmo tempo e a base não estar completa. Os mesmos autores reforçam que caso a base não

seja completada de forma apropriada[14], o método MODI[15] não poderá ser aplicado e propõe uma metodologia para completar a base. A proposta é aplicada para dois exemplos 4x4, entretanto, não existe demonstração de que o método funciona para qualquer PT.

Por outro lado, Silva (2012) descreve um procedimento para completar a base em que existe garantia de que nenhum ciclo será formado entre as variáveis básicas e, consequentemente, o método MODI pode ser aplicado.

Considerando o problema apresentado na figura 14 e sendo os valores das variáveis x_{ij} de cada variável básica apresentados nas células (i, j), observa-se que o problema equilibrado de 5 origens e 5 destinos possui uma solução factível com 6 variáveis diferentes de zero. Entretanto, para que a solução fosse uma solução básica factível seriam necessárias $m + n - 1 = 9$ variáveis básicas.

A base deve ser então completada, utilizando as seis variáveis com valores diferentes de zero e mais três variáveis básicas degeneradas, de forma a não existirem ciclos no quadro, para que assim o método MODI possa ser aplicado. Caso, por exemplo, a variável x_{12} fosse definida como sendo uma variável básica degenerada, formar-se-ia o ciclo $(\mathcal{O}_1, \mathcal{D}_2), (\mathcal{O}_1, \mathcal{D}_5), (\mathcal{O}_5, \mathcal{D}_5), (\mathcal{O}_5, \mathcal{D}_2)$. Assim, observa-se que não é qualquer variável que pode ser alocada para ser básica degenerada.

	1	2	3	4	5	Oferta
1					100	100
2			80			80
3				120		120
4	60					60
5		40			80	120
Demanda	60	40	80	120	180	

FIGURA 14 – EXEMPLO DE SOLUÇÃO PARA UM PT EM QUE É NECESSÁRIO COMPLETAR A BASE COM VARIÁVEIS DEGENERADAS

FONTE: Silva (2012)

A verificação no quadro das variáveis candidatas a serem básicas degeneradas não é imediata, uma vez que nem sempre é fácil a verificação da não

[14] Caso as colunas associadas às variáveis escolhidas para "formar a base" não sejam linearmente independentes, o que é equivalente a existir ciclo no quadro, a solução não será básica.

[15] O método MODI, também conhecido como método u-v, explicado na seção 2.3.8, constitui-se em calcular os valores das variáveis duais pela resolução de um sistema de equações associadas às variáveis básicas e com base neles, os valores de custos atualizados para as variáveis não básicas.

existência de ciclos no quadro. Entretanto, quando a solução apresentada na figura 14 é representada em grafo (figura 15), percebe-se a existência de uma floresta, ou seja, um conjunto de árvores. Foi verificado anteriormente que no caso da inserção da variável x_{12} como básica degenerada implicaria na formação de um ciclo dentro do quadro.

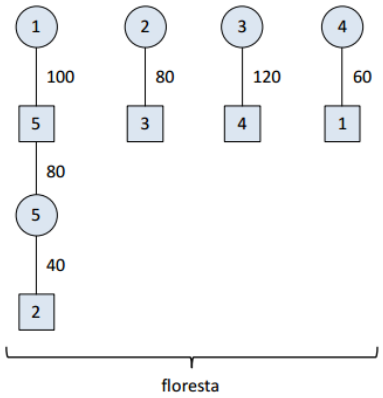


FIGURA 15 – REPRESENTAÇÃO DA FLORESTA ANTES DA BASE SER COMPLETADA
FONTE: Silva (2012)

Ao ser observada a figura 15, constata-se que a origem 1 e o destino 2 estão na mesma árvore. Em Bazaraa, Jarvis e Sherali (2010) é detalhado que a formação de ciclo no grafo implica em ciclo no quadro e vice-versa. Diante disso, sendo $\mathcal{T}_1, \ldots, \mathcal{T}_p$ as árvores que formam a floresta, a base pode ser completada, sucessivamente, pela inclusão de $p-1$ variáveis em que origem e destino pertencem a árvores distintas a cada iteração.

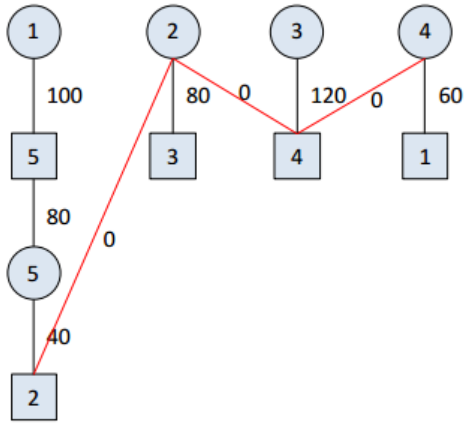


FIGURA 16 – TRANSFORMAÇÃO DA FLORESTA EM ÁRVORE PELA INSERÇÃO DE VARIÁVEIS BÁSICAS DEGENERADAS

FONTE: Silva (2012)

Desta forma, a representação em quadro para a solução desenvolvida na figura 16 é apresentada na figura 17.

	1	2	3	4	5	Oferta
1					100	100
2		0	80	0		80
3				120		120
4	60			0		60
5		40			80	120
Demanda	60	40	80	120	180	

FIGURA 17 – SOLUÇÃO EM QUADRO APÓS A INSERÇÃO DAS VARIÁVEIS BÁSICAS DEGENERADAS

FONTE: Silva (2012)

O fato da representação em árvore tornar mais simples a definição de quais são as possíveis variáveis básicas degeneradas exemplifica a importância que a estrutura utilizada pode ter na facilidade de implementação de um método, especialmente de grande dimensão, assim como no resultado computacional quando o tempo é avaliado.

A importância da existência de procedimentos para transformar uma solução factível em solução básica factível surge pelo fato de métodos de solução inicial poderem gerar soluções com a base incompleta.

2.3.7 O significado do custo atualizado

O princípio básico para a melhora de uma solução atual no Problema de Transporte é determinar se a utilização de um arco que não está sendo utilizado gera redução no custo total.

O custo atualizado de uma variável não básica x_{ij} representa qual será a variação no valor da função objetivo para cada unidade adicionada ao valor desta variável. Quando é alterado o valor da variável x_{ij}, faz-se necessário recalcular os

valores de outras variáveis para que a factibilidade seja mantida. A ideia descrita nos próximos parágrafos foi desenvolvida e apresentada por Charnes e Cooper (1954) e é denominada de método *Stepping Stone*.

No caso particular, não considerando a existência de possíveis variáveis básicas degeneradas, de quando a variável $x_{i_e j_e}$ deixa de ser não básica e torna-se básica, ela passa de um valor zero para algum valor positivo. Para estudar qual o efeito desta mudança na função objetivo, a título de exemplo, será analisado qual o impacto de a variável $x_{i_e j_e}$ passar de 0 para 1. Para isso será considerado um PT que possui a solução atual apresentada na figura 18. A explicação realizada a seguir é baseada em Arenales *et al*. (2007).

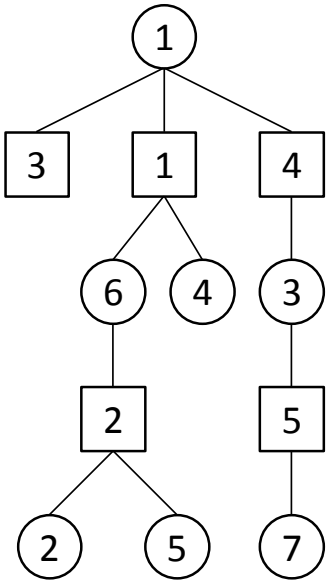


FIGURA 18 – EXEMPLO PARA EXPLICAÇÃO DO SIGNIFICADO DO CUSTO ATUALIZADO
FONTE: O Autor (2014)

Utilizando como exemplo inicial o aumento do valor da variável $x_{5,5}$ de 0 para 1, tem-se que, se nenhuma outra alteração for realizada, a quantidade de itens que serão transportados a partir da origem 5 aumentará em uma unidade, tornando a solução infactível.

Para corrigir a infactibilidade referente à origem 5, poderia ser diminuída em uma unidade a quantidade transportada da origem 5 para o destino 2, ou seja, diminuir em uma unidade o valor da variável $x_{5,2}$. Porém, esta correção para a factibilidade referente à origem 5 gerou uma infactibilidade na restrição referente ao destino 2, o que pode ser corrigido adicionando uma unidade ao valor da variável

$x_{6,2}$, resolvendo o problema de infactibilidade do destino 2, mas gerando problema de factibilidade na origem 6. Continuando o mesmo raciocínio, seria diminuído em uma unidade o valor da variável $x_{6,1}$, adicionada uma unidade ao valor da variável $x_{1,1}$, diminuída uma unidade do valor da variável $x_{1,4}$, adicionada uma unidade ao valor da variável $x_{3,4}$ e, por fim, diminuída uma unidade do valor da variável $x_{3,5}$. Esta última correção resolveria o problema de infactibilidade referente à origem 3 e também o referente ao destino 5, originado pelo aumento do valor da variável $x_{5,5}$.

Desta forma, o processo de acrescentar e diminuir unidades do valor das variáveis no ciclo entre os nós $\{O_5, D_2, O_6, D_1, O_1, D_4, O_3, D_5, O_5\}$ manteve a factibilidade do problema. Por outro lado, como a cada arco $origem - destino$, em que a quantidade transportada é representada pela variável x_{ij}, existe um custo c_{ij} associado, a mudança nos valores das variáveis implicará numa variação do custo total.

Para um aumento de uma unidade no valor da variável $x_{5,5}$, o valor da função objetivo terá um acréscimo de $c_{5,5}$. Acrescentando uma unidade ao valor da variável $x_{5,5}$ e diminuindo em uma unidade o valor da variável $x_{5,2}$, o valor da função objetivo terá um acréscimo de $c_{5,5}$ e um decréscimo de $c_{5,2}$, ou seja, $\bar{z} = z + c_{5,5} - c_{5,2}$. De forma, que após o processo no ciclo $\{O_5, D_2, O_6, D_1, O_1, D_4, O_3, D_5, O_5\}$, o novo valor da função objetivo seria $\bar{z} = z + c_{5,5} - c_{5,2} + c_{6,2} - c_{6,1} + c_{1,1} - c_{1,4} + c_{3,4} - c_{3,5}$.

Assim, nesse exemplo, no caso de $c_{5,5} - c_{5,2} + c_{6,2} - c_{6,1} + c_{1,1} - c_{1,4} + c_{3,4} - c_{3,5} < 0$, tem-se que o aumento do valor da variável $x_{5,5}$ gera redução no valor da função objetivo e que a redução será de $c_{5,5} - c_{5,2} + c_{6,2} - c_{6,1} + c_{1,1} - c_{1,4} + c_{3,4} - c_{3,5}$ para cada unidade aumentada no valor da variável $x_{5,5}$.

Por outro lado, no caso de $c_{5,5} - c_{5,2} + c_{6,2} - c_{6,1} + c_{1,1} - c_{1,4} + c_{3,4} - c_{3,5} > 0$, tem-se que o aumento de $\theta_{5,5}$ unidades no valor da variável $x_{5,5}$ gera um aumento de $\theta_{5,5}\left(c_{5,5} - c_{5,2} + c_{6,2} - c_{6,1} + c_{1,1} - c_{1,4} + c_{3,4} - c_{3,5}\right)$ no valor da função objetivo.

2.3.8 O dual do Problema do Transporte

O dual do Problema de Transporte apresentado em (1)-(4), conforme já apresentado por Murty (1983), Bazarra, Jarvis e Sherali (2010), pode ser dado por:

$$\max w(u,v) = \sum_i u_i a_i + \sum_j v_j b_j \tag{14}$$

$$u_i + v_j \leq c_{ij}, i = 1, \dots, m; j = 1, \dots, n \tag{15}$$

$$u_i, v_j \; quaisquer, i = 1, \dots, m; j = 1, \dots, n \tag{16}$$

Onde u_i e v_j representam as variáveis duais. Além disso, pelo fato de as restrições do PT serem de igualdade, u_i e v_j são variáveis livres.

O teorema da folga complementar implica que se a variável x_{ij} for básica, então $c_{ij} - u_i - v_j = 0$. Como são $m + n - 1$ variáveis básicas, tem-se um sistema de equações lineares com $m + n - 1$ equações e $m + n$ variáveis. Desta forma, uma variável recebe um valor qualquer, 0 por exemplo, e as demais são obtidas por retro-substituição. Usualmente, atribui-se o valor 0 para a variável u_1. (BAZARAA, JARVIS e SHERALI 2010)

Caso os custos c_{ij} sejam valores inteiros, isto é, $c_{ij} \in \mathbb{Z}$, e seja definido um valor inteiro para uma variável dual, ao final do processo de retro-substituição todos os valores das variáveis duais serão inteiros. Este resultado é trivial, uma vez que todas as variáveis serão calculadas pela diferença de valores inteiros.

Uma vez obtidos os valores das variáveis duais, os custo atualizados para cada variável não básica pode ser calculado por $\bar{c}_{ij} = c_{ij} - u_i - v_j$. Conforme citado por Murty (1983), a ausência de custo atualizado negativo é suficiente para a otimalidade do problema. Ou seja, dada uma solução básica factível, caso todos os custos atualizados sejam maiores ou iguais a zero, esta solução será uma solução ótima para o problema. Desta forma, tem-se um método para avaliar otimalidade com base apenas numa solução básica atual.

O procedimento de calcular as variáveis duais e assim obter os custos atualizados sem a necessidade da realização do *Stepping Stone* foi apresentado por Reinfeld e Vogel (1958) e denominado Método MODI (*Modified Distribution*). O

mesmo método é também conhecido por método $u - v$, em função do cálculo das variáveis duais.

A maior diferença entre o *Stepping Stone* e o MODI está no procedimento utilizado para o teste de otimalidade. No primeiro, identifica-se o $\theta - loop$ para cada variável não básica, calcula-se o custo atualizado de cada uma delas e escolhe-se para entrar na base a que possui o custo atualizado mais negativo[16]. Por outro lado, no segundo, utilizam-se as variáveis duais (também chamadas de potenciais ou multiplicadores) para calcular o custo atualizado de cada variável não básica, escolhe-se a de custo atualizado mais negativo para entrar na base e, para somente esta, identifica-se o $\theta - loop$.

2.3.9 Mudança de solução básica

Para a mudança de uma solução básica factível atual para uma nova é necessário que uma variável não básica torne-se básica, uma básica torne-se não básica e que nenhuma variável básica assuma valor negativo, além de satisfazer as restrições de oferta e demanda. Estas garantias são obtidas por um método de acréscimos e decréscimos nos valores de variáveis ao longo de um ciclo, o $\theta - loop$. O método é amplamente divulgado na literatura, como em Murty (1983), Bazaraa, Jarvis e Sherali (2010).

Para a implementação computacional do método é necessário um algoritmo para a identificação do $\theta - loop$, seja na estrutura de quadro ou árvore.

Souza (2004) descreve um algoritmo simples de seis passos, denominado algoritmo do ciclo único, para identificação do ciclo no quadro.

1. Marcar todas as variáveis básicas e a variável que entra na base como pertencentes ao ciclo único;
2. Para cada linha do quadro de transporte, contar o número de variáveis pertencentes ao ciclo único;
3. Se a linha tiver apenas uma variável pertencente ao ciclo único, excluir a variável do ciclo único e marcar que a linha foi avaliada;

[16] Qualquer variável que possui custo atualizado negativo, desde que a variável que está saindo da base não seja degenerada, gera melhoria no valor da função objetivo. A escolha da variável que possui o custo atualizado mais negativo busca diminuir o número de iterações para a resolução do problema.

4. Para cada coluna do quadro de transporte, contar o número de variáveis pertencentes ao ciclo único;
5. Se a coluna tiver apenas uma variável pertencente ao ciclo, excluir a variável do ciclo único e marcar a coluna que foi avaliada;
6. Se o número de variáveis, em cada uma das linhas e em cada uma das colunas, pertencentes ao ciclo único for zero ou dois, parar; caso contrário, voltar ao passo 2.

No presente trabalho, para a identificação do ciclo, utiliza-se uma estrutura semelhante às encontradas em O'Connor (2002), Papamanthou *et al.* (2004) e Silva (2012). Para o problema da figura 19, quando a variável x_{67} torna-se básica, o ciclo é formado pelos arcos associados às variáveis $x_{67}, x_{27}, x_{22}, x_{12}, x_{15}, x_{55}, x_{53}, x_{33}, x_{36}$ e x_{66}.

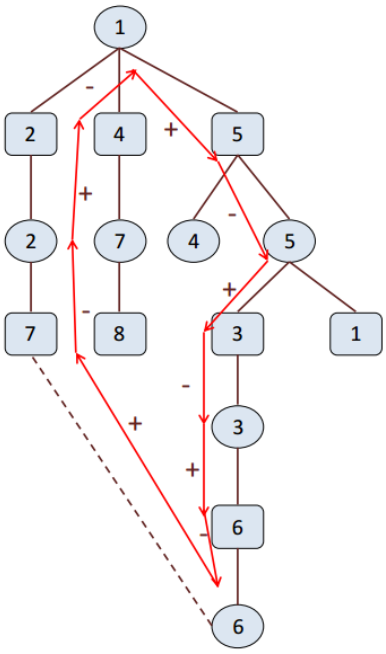


FIGURA 19 – EXEMPLO PARA ENCONTRAR O $\theta - loop$ EM GRAFO
FONTE: Silva (2012)

O mesmo ciclo da figura 19 pode ser verificado também na figura 20. Este exemplo mostra a dificuldade visual de encontrar o $\theta - loop$ no quadro.

	1	2	3	4	5	6	7	8	Oferta
1		x		x	x				
2		x					x		
3			x			x			
4					x				
5	X		X		x				
6						x	θ		
7				x				x	
Demanda									

FIGURA 20 – EXEMPLO DO $\theta - loop$ EM QUADRO
FONTE: Silva (2012)

As variáveis pertencentes ao ciclo encontrado tem os seus respectivos valores atualizados, sendo realizadas adições e subtrações, alternadamente, ao longo do ciclo. O valor que a variável x_{67} assume é o maior valor possível para que nenhuma variável torne-se negativa.

2.4 GRAFOS BIPARTIDOS, PROBLEMA DE TRANSPORTE E ÁRVORES GERADORAS

A representação do Problema de Transporte em grafo é um exemplo de grafo bipartido, pois tem-se dois conjuntos de vértices distintos, origens e destinos, e as conexões ocorrem somente de um conjunto para outro.

Como cada solução básica factível para o PT pode ser representada por uma árvore geradora e o processo de mudança de base é análogo ao movimento no grafo das árvores geradoras para uma árvore geradora adjacente, são aqui apresentadas características das árvores geradoras em grafos bipartidos.

Quando tem-se um grafo completo com m vértices, o número de árvores geradoras possíveis é m^{m-2}, sendo uma das provas existentes para esta propriedade realizada pela codificação de Prüfer.

Já para o caso de grafos bipartidos no qual um conjunto possui m vértices e outro n vértices, o número de árvores geradoras é $m^{n-1}n^{m-1}$. Desta forma, para um PT com quatro origens e dois destinos, como o da figura 21, o número de árvores geradoras é $2^3 4^1 = 32$. Cada solução básica factível é representada por uma árvore

geradora, mas nem toda árvore geradora está associada a uma solução básica factível.

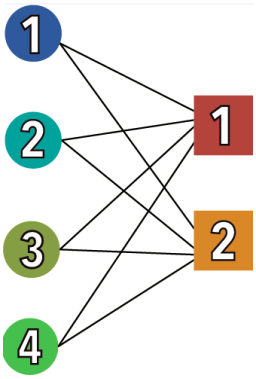


FIGURA 21 – EXEMPLO DE UM GRAFO BIPARTIDO $K_{4,2}$
FONTE: O Autor (2014)

As 32 árvores geradoras para o grafo da figura 21 são apresentadas na figura 22. Para este exemplo optou-se por um padrão de desenho diferente das demais árvores neste trabalho para facilitar a identificação, uma vez que na figura 22 são apresentadas 32 árvores.

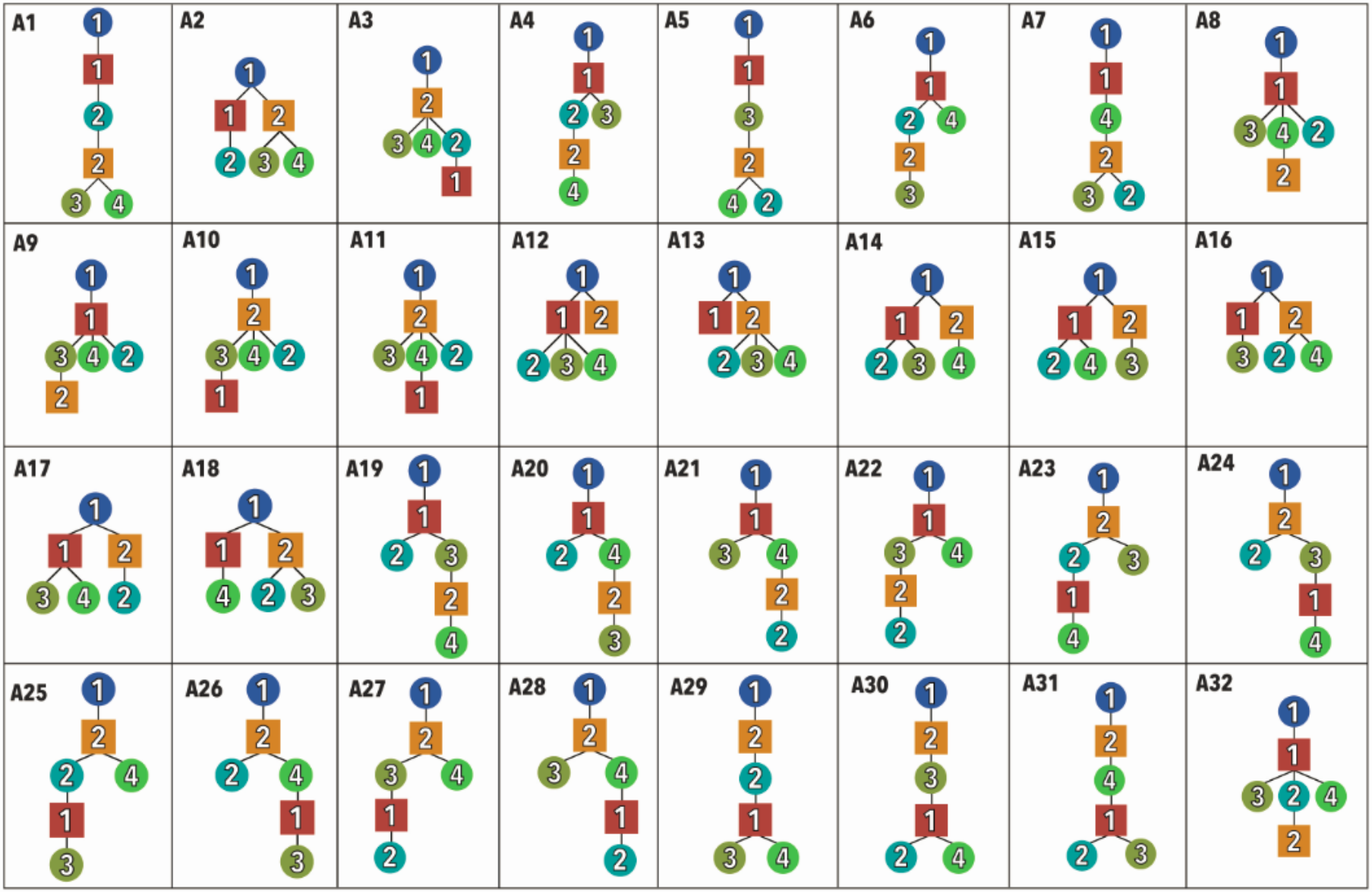


FIGURA 22 – ENUMERAÇÃO DAS ÁRVORES GERADORAS PARA UM GRAFO $K_{4,2}$
FONTE: O Autor (2014)

Uma representação do grafo de árvores para esta situação resultaria numa figura confusa e, portanto, é aqui apresentada somente a matriz de adjacências (figura 23) destas árvores.

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20	A21	A22	A23	A24	A25	A26	A27	A28	A29	A30	A31	A32
A1	0	1	1	1	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
A2	1	0	1	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
A3	1	1	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	0	0	0
A4	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	1
A5	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0
A6	1	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	0	0	0	0	1
A7	1	0	0	0	1	1	0	0	0	0	1	0	1	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0
A8	0	0	0	1	0	0	0	0	1	0	0	1	0	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1	1
A9	0	0	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0	0	0	0	0	0	1	0	1
A10	0	0	1	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0
A11	0	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	1	1	0	1	0	1	0	0	0	0
A12	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
A13	1	1	1	0	1	0	1	0	0	1	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A14	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	1	0	0	0	1	0
A15	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	1	0	1	0	0
A16	0	0	0	1	1	0	0	0	0	1	0	0	1	1	0	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0
A17	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	1	1	0	1	0	1	0	0	1	0	0	1
A18	0	0	0	0	0	1	1	0	0	0	1	0	1	0	1	0	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
A19	1	1	0	1	1	0	0	1	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0
A20	1	1	0	0	0	1	1	1	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
A21	0	0	0	1	1	0	1	1	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1
A22	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	1	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1
A23	0	0	1	0	0	1	0	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	1	1	1	0	0
A24	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	1	1	0	0	1	0	0	1	1	0	0
A25	0	0	1	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1	0	1	0	1	0
A26	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	1	0	0	1	1	0	0	0	1	0	1	0
A27	0	1	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	1	1	0
A28	0	1	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	1	0	0	1	1	0
A29	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1
A30	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	1	0	0	1	1	1	0	1	0
A31	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0
A32	0	0	0	1	0	1	0	1	1	0	0	1	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0

FIGURA 23 – MATRIZ DE ADJACÊNCIA PARA AS ÁRVORES GERADORAS DA FIGURA 22
FONTE: O Autor (2014)

Porém, duas árvores adjacentes não representam necessariamente duas bases vizinhas para o PT. Por exemplo, a árvore A21 da figura 22 pode ser obtida pela inclusão da aresta $(4,1)$ no lugar da aresta $(3,2)$ da árvore A5 da figura 22. Entretanto, caso a árvore A21 representasse uma solução básica factível e a variável x_{41} fosse entrar na base, as candidatas a deixarem a base seriam as variáveis x_{42} e x_{31} e, consequentemente, poderia somente ser eliminado o arco correspondente a uma destas duas variáveis. Desta forma, para o caso do grafo de

árvores representando soluções básicas factíveis para o PT, as árvores A21 e A5 não seriam árvores adjacentes. Entretanto, seria possível chegar na árvore A21 a partir da árvore A5 pelo caminho A21-A22-A5, conforme representado na figura 24.

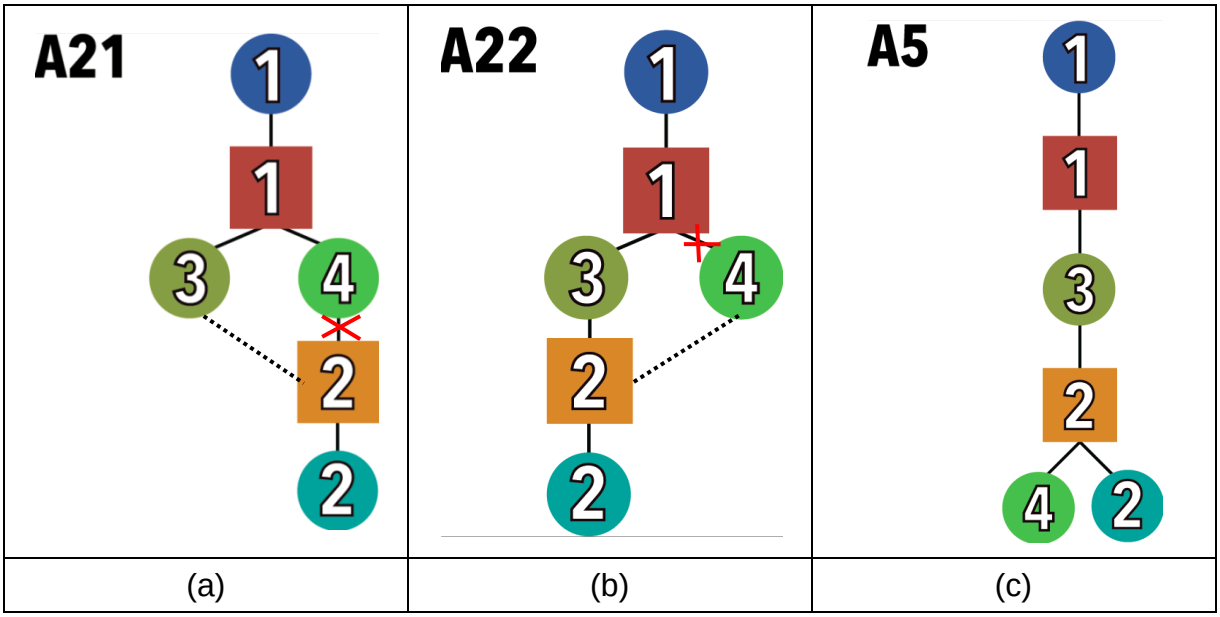


FIGURA 24 – EXEMPLO DE CAMINHO ENTRE ÁRVORES ADJACENTES PARA O PROBLEMA DE TRANSPORTE
FONTE: O Autor (2014)

Poderia ainda ser criada a matriz de adjacência das árvores geradoras para o Problema de Transporte e, com ela e a utilização de um algoritmo de caminho mínimo, seria possível obter o número de mínimo de troca de arestas para a partir de uma árvore T_i conseguir chegar numa árvore T_j por um caminho de árvores.

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20	A21	A22	A23	A24	A25	A26	A27	A28	A29	A30	A31	A32
A1	0	1	1	1	1	1	1	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
A2	1	0	**0**	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	**0**	**0**	0	0	0	0	0	0	1	1	0	0	0	0
A3	1	**0**	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	**0**	**0**	0	0	0	0
A4	1	0	0	0	**0**	0	0	**0**	0	0	0	0	0	1	0	**0**	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	1

FIGURA 25 – PARTE DE MATRIZ DE DISTÂNCIA ENTRE ÁRVORES DO PROBLEMA DE TRANSPORTE PARA AS ÁRVORES GERADORAS DA FIGURA 22
FONTE: O Autor (2014)

Na figura 25 é apresentado como exemplo parte da matriz da figura 24 adaptada para o caso de adjacência de árvores geradoras para o Problema de Transporte, sendo destacados os casos em que existia adjacência no grafo de árvores, mas não para o Problema de Transporte.

Além disso, as restrições de ofertas e demandas alteram o número de árvores factíveis e, como consequência, a matriz de adjacência e também a matriz de distância entre as árvores, uma vez que no $\theta - loop$ formado pela inclusão de uma variável na base não é qualquer arco que pode ser removido.

Desta forma, ainda em menor quantidade são as árvores adjacentes quando consideradas as restrições de ofertas e demandas e, consequentemente, a distância entre cada par de árvores (soluções básicas factíveis) torna-se maior. Entretanto, mesmo que a distância entre árvores forneça um limite inferior do número de iterações necessárias para que a partir de uma solução básica factível inicial seja encontrada a solução ótima, não existe qualquer tipo de garantia de que com menos soluções básicas factíveis sejam necessárias menos iterações.

Para um PT de cinco origens e cinco destinos o número de árvores geradoras é $5^4 5^4 = 390.625$. Considerando um exemplo no qual as ofertas de cada origem e as demandas de cada destino são de 50 unidades, o número de árvores geradoras factíveis é de 240.000, já quando o vetor de ofertas é $a = \{50, 20, 90, 10, 130\}$ e o vetor de demandas é $b = \{50, 50, 50, 40, 110\}$ o número de árvores geradoras factíveis diminui para 13.293. Ressalta-se que estes valores foram obtidos pela resolução dos sistemas associados a cada base do problema.

Se por um lado diferentes vetores de valores de ofertas e demandas podem aumentar a distância mínima entre duas árvores geradoras factíveis (soluções básicas factíveis), por outro um número menor de vértices pode fazer com que a solução seja, na média, encontrada com menos iterações.

Conforme a dimensão do problema cresce, aumenta de forma exponencial o número de árvores geradoras, tornando não viável no presente momento um estudo sobre o caminho seguido no grafo de árvores entre a solução inicial e a solução ótima.

Embora não seja estudado aqui com mais detalhes o comportamento do algoritmo no grafo de árvores, optou-se por escrever esta seção pelo fato de os conceitos aqui apresentados terem sido úteis para entender a complexidade de resolução para o PT, no que se refere ao número de iterações para resolução dependendo da solução inicial e diferentes valores de ofertas e demandas.

2.5 NOTA SOBRE COMPLEXIDADE DE RESOLUÇÃO DO PROBLEMA DE TRANSPORTE

Um estudo sobre a complexidade do Problema de Transporte é apresentado, entre outros, em Bazaraa, Jarvi e Sherali (2010). Porém, o tempo de resolução, quando o algoritmo é implementado computacionalmente, não depende apenas da ordem de complexidade teórica.

O simplex revisado é um exemplo de adaptação de um método que permite a obtenção da resposta com um menor esforço computacional em relação ao original. Conforme descrito por Papadimitriou e Steiglitz (1982), no cálculo em quadro é necessário atualizar $(m+1)x(n+1)$ valores a cada iteração, enquanto no simplex revisado uma matriz[17] de $(m+1)x(m+1)$ é atualizada a cada iteração.

Os mesmos autores comentam também que a vantagem do simplex revisado não está propriamente no número de atualizações necessárias, uma vez que o simplex revisado necessita de cálculos de produtos internos para a obtenção dos custos atualizados. As vantagens citadas são:

1. Não é necessário calcular todos os custos atualizados. A variável não básica associada ao primeiro custo atualizado negativo pode ser escolhida para tornar-se básica e ocorrerá redução no valor da função objetivo.
2. As operações para o cálculo dos custos atualizados utilizam as colunas originais dos coeficientes das variáveis nas restrições. Como em diversos problemas a matriz formada por estas colunas é esparsa, os custos atualizados podem então ser calculados de forma rápida.
3. A esparsidade da matriz original dos coeficientes das variáveis nas restrições permite que as informações sejam armazenadas numa melhor estrutura de dados.

Dado um mesmo problema de programação linear e quando utilizados os mesmos critérios para entrada e saída de variáveis da base, tanto a resolução pelo simplex em quadro como a resolução pelo simplex revisado percorrem exatamente

[17] Como o número n de variáveis é geralmente muito maior do que o número m de restrições, a matriz a ser atualizada torna-se menor.

os mesmos vértices a cada iteração. Porém, computacionalmente o cálculo por meio do simplex revisado gera soluções em tempos menores do que quando utilizado o simplex em quadro. Ou seja, o tempo computacional de resolução de um problema de programação linear não depende apenas da complexidade matemática teórica, mas também da sua implementação.

Quando não é possível analisar cada caso de um determinado problema, uma alternativa é analisar a complexidade do algoritmo pelo pior caso. O problema é que nem sempre a complexidade de resolução no pior caso é similar a complexidade no caso médio. O próprio método simplex, conforme citado por Murty (1983) é um caso desta situação.

De forma geral, a complexidade de um algoritmo pode ser analisada pelo número de operações realizadas a cada iteração e o número de iterações para a resolução do mesmo. Para o caso do PT, considerando em particular a implementação realizada no presente trabalho, conforme descrita no capítulo 4, o número de operações no caso médio é distante do número de operações para o pior caso.

Desta forma, na presente tese não é realizada uma análise teórica sobre a complexidade dos algoritmos de resolução do Problema de Transporte, mas são realizados comentários sobre os números de operações necessárias nos principais passos do algoritmo implementado.

3 RECÁLCULO DOS CUSTOS ATUALIZADOS NA MUDANÇA DE BASE

Para a resolução de um Problema de Transporte, o tempo para o cálculo dos novos valores de custos atualizados para as variáveis não básicas é responsável por parcela do tempo de processamento. Conforme já citado anteriormente, o método MODI proporciona economia de tempo computacional em relação ao método *Stepping Stone*. Além disso, a estrutura de dados utilizada na implementação é decisiva para o desempenho.

Em trabalhos como o de Silva (2012), observou-se a vantagem da utilização de uma estrutura de árvore em relação a estrutura de quadro. O principal ganho ocorre na identificação do $\theta - loop$ para a definição de qual variável deixa de ser básica.

No trabalho de Silva (2012) verificou-se ainda que o maior ganho de tempo computacional para problemas esparsos ocorreu devido a possibilidade de calcular uma menor quantidade de custos atualizados. Diante disso, surge o interesse também em buscar respostas para as seguintes perguntas:

- É necessário recalcular todas as variáveis duais a cada iteração?
- É necessário recalcular todos os custos atualizados a cada iteração?
- Para os custos atualizados que necessitam ser recalculados, a maneira mais simples para o recálculo é a do método MODI?

Quando o PT é estudado em quadro, surge a impressão de que a resposta para todas estas perguntas é sim. Porém, quando o problema é analisado em estrutura de árvore, podem ser identificadas algumas simplificações.

3.1 CÁLCULO DOS NOVOS CUSTOS ATUALIZADOS APÓS A ITERAÇÃO

Na presente seção é apresentada a demonstração sobre os novos valores de custos atualizados após cada iteração. Para exemplificar conceitos utilizados na demonstração, é utilizada como exemplo a árvore representada na Figura 26.

As árvores geradoras associadas à soluções básicas factíveis do PT possuem arcos, ou seja, as ligações entre os vértice são direcionadas. Desta forma,

seria mais rigoroso dizer a cadeia entre dois vértices e representar as ligações com setas para indicar o sentido da origem para o destino. Entretanto, na maior parte dos artigos pesquisados sobre o PT, especialmente sobre o PTCF, é utilizada a expressão caminho entre dois vértices, mesmo utilizando o termo arco. Para manter esta terminologia e não causar dúvidas, as árvores serão desenhadas com arestas, sem direção, e poderá ser entendido qual o sentido pelo fato de os vértices referentes às origens serem representados por círculos e os referentes aos destinos representados por quadrados.

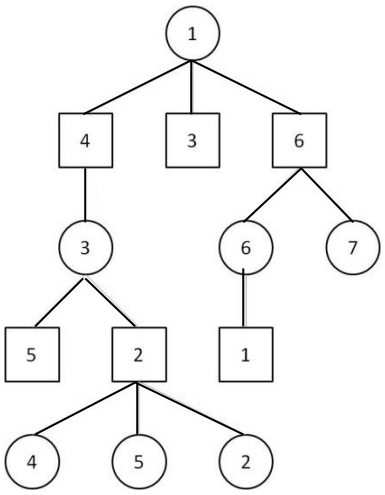


FIGURA 26 – EXEMPLO DE ÁRVORE PARA O PROBLEMA DO TRANSPORTE
FONTE: O Autor (2014)

Seja $\mathcal{T}$ a árvore associada a solução básica factível atual $\mathcal{B}$ do problema e sejam $s, t \in \mathcal{T}$, então $\xi(s,t)$, no presente trabalho, denota o ancestral comum entre s e t de maior profundidade. Por exemplo, para a árvore representada na figura 26, tem-se $\xi(\mathcal{O}_4, \mathcal{D}_5) = \mathcal{O}_3$, $\xi(\mathcal{D}_2, \mathcal{O}_7) = \mathcal{O}_1$ e $\xi(\mathcal{O}_3, \mathcal{D}_5) = \mathcal{O}_3$.

Considerando a árvore $\mathcal{T}$ associada à solução básica factível atual $\mathcal{B}$, é possível realizar o cálculo dos valores das variáveis duais por meio de uma busca em largura (figura 27) em $\mathcal{T}$, o que é equivalente ao cálculo por retro-substituição no sistema de equações resultante do teorema das folgas complementares.

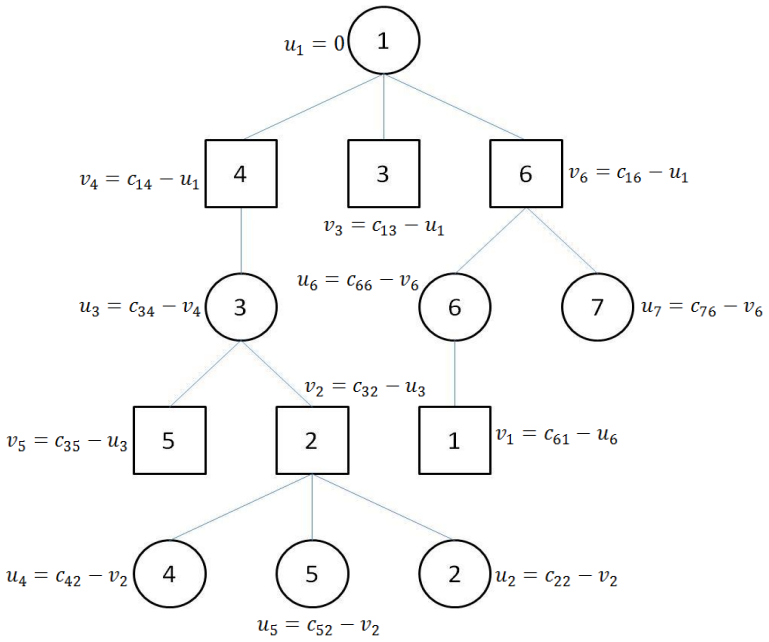


FIGURA 27 – EXEMPLO DE CÁLCULOS DAS VARIÁVEIS DUAIS POR MEIO DE UMA BUSCA EM LARGURA NA ÁRVORE

FONTE: O Autor (2014)

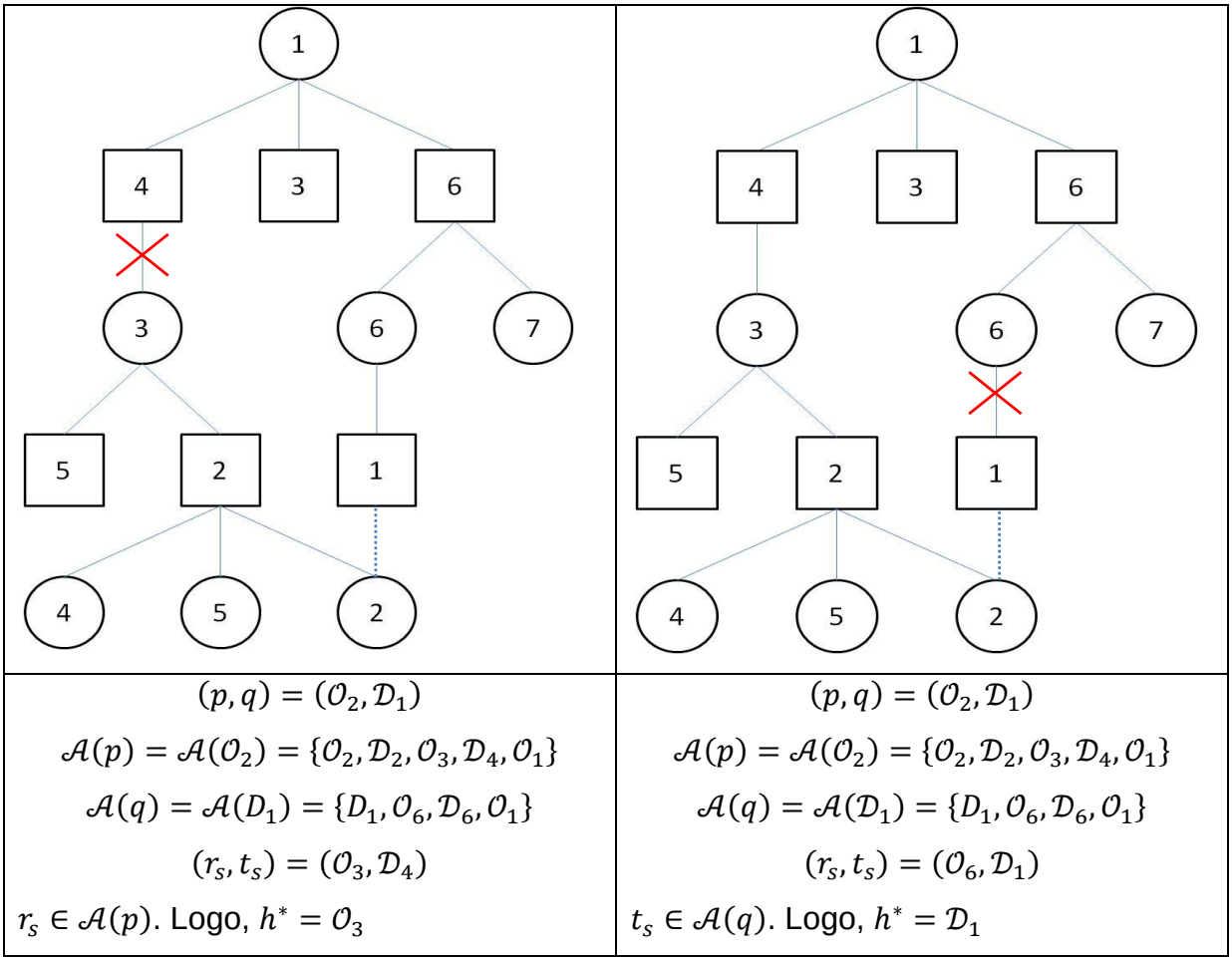
Para o cálculo das variáveis duais utilizando a estrutura de árvore, define-se inicialmente o valor da variável dual associada ao nó raiz, em seguida calcula-se o valor das variáveis duais associadas a cada filho do nó raiz. O processo repete-se iterativamente por busca em largura até que todas as variáveis duais tenham sido calculadas. Ressalta-se que o processo também poderia ser realizado por uma busca em profundidade.

A variável não básica x_{pq} tornar-se básica é equivalente ao arco (p, q) que está fora da árvore passar a fazer parte dela. Por outro lado, a variável básica $x_{r_s t_s}$ tornar-se não básica é equivalente ao arco (r_s, t_s) que está na árvore ser excluído dela. Este processos, por simplificação de notação, serão chamados de (p, q) entrar na base e (r_s, t_s) sair da base, respectivamente.

A cada iteração do simplex, ocorre o processo de pivoteamento com (p, q) entrando na base e (r_s, t_s) saindo da base. Além disso, a "variável" que deixa de ser básica está no caminho entre p e q, existindo duas possibilidades: a primeira é a de estar no caminho de p a $\xi(p, q)$ e então $r_s \in \mathcal{A}(p)$, já a segunda é a de estar no caminho de q a $\xi(p, q)$ e então $t_s \in \mathcal{A}(q)$. Desta forma, define-se h^* como sendo:

$$h^* = \begin{cases} r_s, se\ r_s \in \mathcal{A}(p) \\ t_s, se\ t_s \in \mathcal{A}(q) \end{cases} \quad (17)$$

Na figura 34 são ilustrados exemplos de cada uma das possibilidades para h^* quando a $(2,1)$ entra na base.



$(p,q) = (\mathcal{O}_2, \mathcal{D}_1)$ $\mathcal{A}(p) = \mathcal{A}(\mathcal{O}_2) = \{\mathcal{O}_2, \mathcal{D}_2, \mathcal{O}_3, \mathcal{D}_4, \mathcal{O}_1\}$ $\mathcal{A}(q) = \mathcal{A}(D_1) = \{D_1, \mathcal{O}_6, \mathcal{D}_6, \mathcal{O}_1\}$ $(r_s, t_s) = (\mathcal{O}_3, \mathcal{D}_4)$ $r_s \in \mathcal{A}(p)$. Logo, $h^* = \mathcal{O}_3$	$(p,q) = (\mathcal{O}_2, \mathcal{D}_1)$ $\mathcal{A}(p) = \mathcal{A}(\mathcal{O}_2) = \{\mathcal{O}_2, \mathcal{D}_2, \mathcal{O}_3, \mathcal{D}_4, \mathcal{O}_1\}$ $\mathcal{A}(q) = \mathcal{A}(\mathcal{D}_1) = \{D_1, \mathcal{O}_6, \mathcal{D}_6, \mathcal{O}_1\}$ $(r_s, t_s) = (\mathcal{O}_6, \mathcal{D}_1)$ $t_s \in \mathcal{A}(q)$. Logo, $h^* = \mathcal{D}_1$

FIGURA 28 – EXEMPLOS DE POSSIBILIDADE PARA h^* PARA A ENTRADA DE $(2,1)$ NA BASE
FONTE: O Autor (2014)

Novamente, utilizando a figura 26 como exemplo, um outro caso seria $(p,q) = (\mathcal{O}_2, \mathcal{D}_6)$ e existiriam três possibilidades para (r_s, t_s), a saber: $(2,2)$, $(3,4)$ e $(1,6)$, sendo os três casos representados na figura 29.

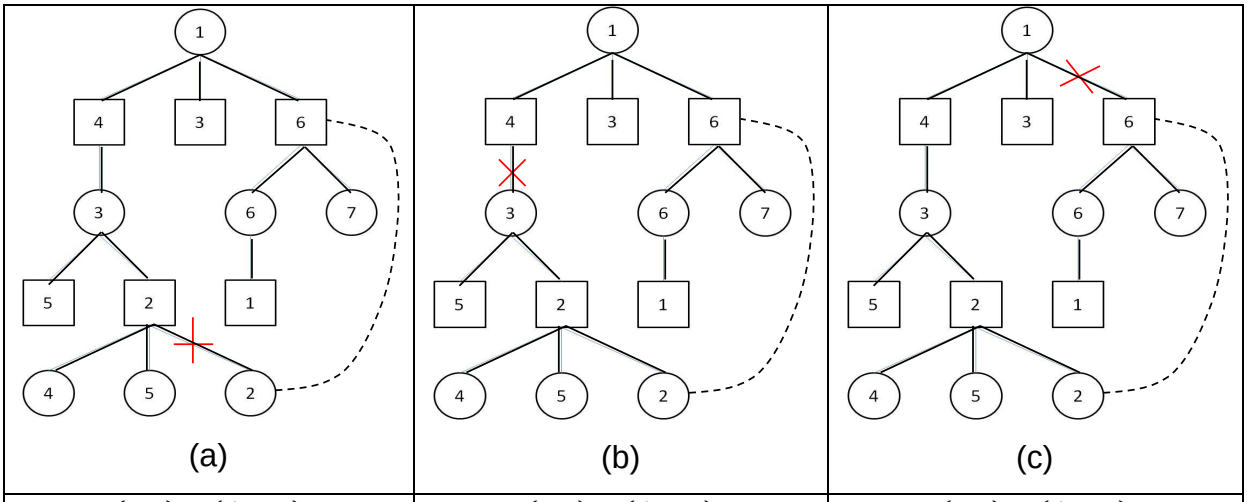


FIGURA 29 – EXEMPLOS DE POSSIBILIDADE PARA h^* PARA A ENTRADA DE $(2,6)$ NA BASE
FONTE: O Autor (2014)

No caso da figura 29 (b) em que $(p,q) = (\mathcal{O}_2, \mathcal{D}_6)$ e $(r_s, t_s) = (\mathcal{O}_3, \mathcal{D}_4)$, tem se que $r_s = \mathcal{O}_3 \in \mathcal{A}(\mathcal{O}_2)$. Logo $h^* = \mathcal{O}_3$ e a árvore $\mathcal{T}$ associada à solução básica factível antes do pivoteamento é dividida em duas subárvores, $\mathcal{H}$ e $\mathcal{T} - \mathcal{H}$, onde $\mathcal{H}$ é a subárvore formada pelos descendentes de h^* (conforme representação na figura 30).

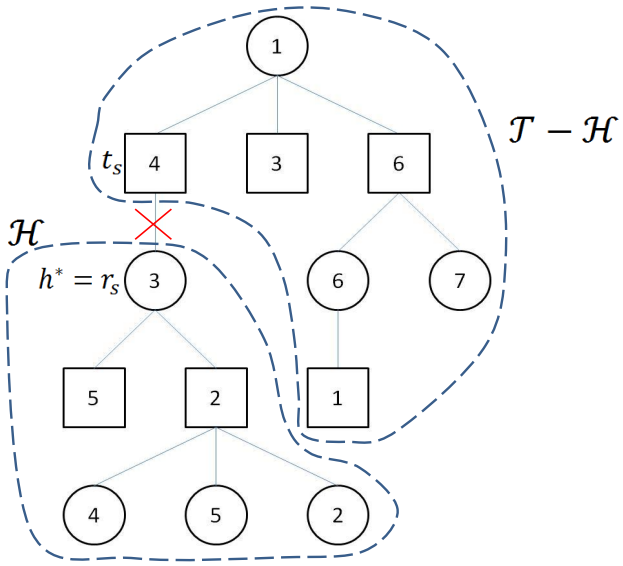


FIGURA 30 – DIVISÃO DA ÁRVORE $\mathcal{T}$ DA FIGURA 26 NAS SUBÁRVORES $\mathcal{H}$ E $\mathcal{T} - \mathcal{H}$
FONTE: O Autor (2014)

De forma separada, as subárvore $\mathcal{H}$ e $\mathcal{T} - \mathcal{H}$ são apresentadas, respectivamente, nas figuras 31 e 32.

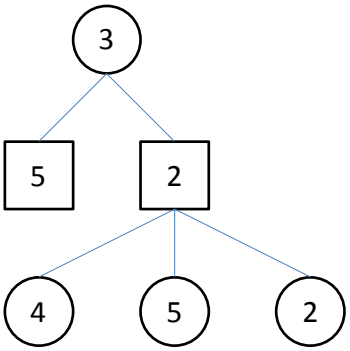


FIGURA 31 – SUBÁRVORE $\mathcal{H}$ PARA O EXEMPLO DA FIGURA 33, COM $(p, q) = (O_2, D_6)$ e $(r_s, t_s) = (O_3, D_4)$

FONTE: O Autor (2014)

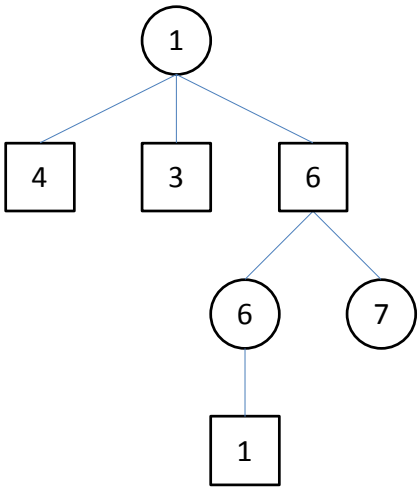


FIGURA 32 – SUBÁRVORE $\mathcal{T} - \mathcal{H}$ PARA O EXEMPLO DA FIGURA 29 (b), COM $(p, q) = (O_2, D_6)$ e $(r_s, t_s) = (O_3, D_4)$

FONTE: O Autor (2014)

Após o pivoteamento, para este exemplo, a subárvore $\mathcal{H}$ deixa de ser enraizada em $h^* = \mathcal{O}_3$ e passa a ser enraizada em $p = \mathcal{O}_2$ e denominada $\mathcal{H}'$ (figura 33). A subárvore $\mathcal{H}'$ é então conectada a subárvore $\mathcal{T} - \mathcal{H}$ (figura 32) por meio da inclusão do arco (p, q), formando a árvore $\mathcal{T}'$ (figura 34).

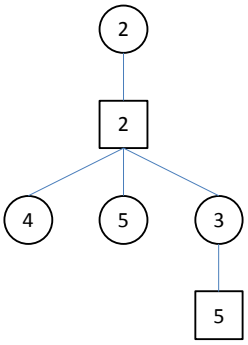


FIGURA 33 – REPRESENTAÇÃO DA SUBÁRVORE $\mathcal{H}'$

FONTE: O Autor (2014)

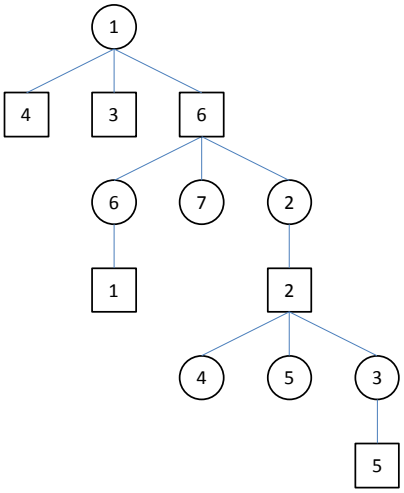


FIGURA 34 – REPRESENTAÇÃO DA ÁRVORE $\mathcal{T}'$

FONTE: O Autor (2014)

Desta forma, tem-se que após (p, q) entrar na base no lugar de (r_s, t_s), a árvore que representa a solução básica atual deixa de ser $\mathcal{T}$ e passa a ser $\mathcal{T}'$.

Para o exemplo da árvore $\mathcal{T}'$ da figura 34, tem-se que para os vértices $V = \{\mathcal{O}_1, \mathcal{D}_4, \mathcal{D}_3, \mathcal{D}_6, \mathcal{O}_6, \mathcal{O}_7, \mathcal{D}_1\}$, que são os vértices pertencentes a árvore $\mathcal{T} - \mathcal{H}$, não ocorrem alterações nos valores das variáveis duais.

De forma genérica tem-se que sendo $u_i, i = 1, \dots, m$ e $v_j, j = 1, \dots, n$ os valores potenciais de cada vértice (ou valores duais associados a cada vértice) antes do pivoteamento e $u_i', i = 1, \dots, m$ e $v_j', j = 1, \dots, n$ os valores potenciais de cada vértice (ou valores duais associados a cada vértice) após o pivoteamento, percebe-se que para $i, j \in \mathcal{T} - \mathcal{H}$, $u_i' = u_i$ e $v_j' = v_j$. Esta verificação visual é possível quando analisada a estrutura em árvore, mas não quando utilizada a usual estrutura em quadro.

Desta forma, identifica-se uma primeira oportunidade de redução de tempo computacional em relação ao método MODI na forma que é apresentado na literatura, pois para $i,j \in \mathcal{T}-\mathcal{H}$ tem-se que $\bar{c}'_{ij} = c_{ij} - u_i' - v_j' = c_{ij} - u_i - v_j = \bar{c}_{ij}$.

Já para o caso de $i,j \in \mathcal{H}$, ocorrem alterações nos valores dos potenciais, mas não é detalhada na presente tese esta alteração porque o objetivo é realizar uma demonstração, sem utilizar os valores das variáveis duais, sobre os novos valores de custos atualizados. De forma que até mesmo para o caso (do parágrafo anterior) em que $i,j \in \mathcal{T}-\mathcal{H}$ é realizada (**Caso 1**) a análise no ciclo sem a utilização das variáveis duais.

O objetivo agora é mostrar, de forma generalizada, qual o custo atualizado de uma variável não básica associada ao arco (i,j) após (p,q) entrar na base no lugar de (r_s,t_s). Para simplificar a notação na demonstração, serão ainda consideradas as seguintes expressões.

$$\hat{\xi} = \xi(i,j) \tag{18}$$

$$\bar{\xi} = \xi(p,q) \tag{19}$$

$$\tilde{\xi} = \begin{cases} \xi(p,j), se\ i \notin \mathcal{H}, j \in \mathcal{H}\ e\ h^* = r_s \\ \xi(q,j), se\ i \notin \mathcal{H}, j \in \mathcal{H}\ e\ h^* = t_s \\ \xi(i,p), se\ i \in \mathcal{H}, j \notin \mathcal{H}\ e\ h^* = r_s \\ \xi(i,q), se\ i \in \mathcal{H}, j \notin \mathcal{H}\ e\ h^* = t_s \end{cases} \tag{20}$$

$$\tilde{\alpha} = \begin{cases} 1, se\ \tilde{\xi} \in \mathcal{D} \\ -1, se\ \tilde{\xi} \in \mathcal{O} \end{cases} \tag{21}$$

$$\bar{\alpha} = \begin{cases} 1, se\ \bar{\xi} \in \mathcal{D} \\ -1, se\ \bar{\xi} \in \mathcal{O} \end{cases} \tag{22}$$

$$\hat{\alpha} = \begin{cases} 1, se\ \hat{\xi} \in \mathcal{D} \\ -1, se\ \hat{\xi} \in \mathcal{O} \end{cases} \tag{23}$$

$$[s,t] = \begin{cases} (s,t), se\ s \in \mathcal{O}\ e\ t \in \mathcal{D} \\ (t,s), se\ s \in \mathcal{D}\ e\ t \in \mathcal{O} \end{cases} \tag{24}$$

Seja $Y = P(s,t)$ para $s \neq t$ em que $s,t \in \mathcal{K}$ e y_i o i-ésimo elemento de Y. Além disso, seja $\omega_i = e_{y_i,y_{i+1}} = [y_i, y_{i+1}], i = 1, \dots, |Y| - 1$. Isto é, ω_i é a i-ésima aresta no caminho de s a t. O conjunto $\Omega(s,t)$, no presente trabalho, denota o conjunto das arestas do caminho de s a t.

Por exemplo, na árvore representada na figura 26, $Y = P(O_5, D_4) = \langle O_5, D_2, O_3, D_4 \rangle$ e $\omega_1 = [O_5, D_2] = (O_5, D_2)$, $\omega_2 = [D_2, O_3] = (O_3, D_2)$ e $\omega_3 = [O_3, D_4] = (O_3, D_4)$. Assim, $\Omega(O_5, D_4) = \{(O_5, D_2), (O_3, D_2), (O_3, D_4)\}$.

Seja c_{ω_k} o custo associado a aresta ω_k, então define-se, no presente trabalho, o custo acumulado no caminho de s a t, $M(s,t)$, por

$$M(s,t) = \sum_{k=1}^{|\Omega|} (-1)^k c_{\omega_k} \tag{25}$$

Desta forma, para o exemplo da figura 33, $M(O_5, D_4) = -c_{(O_5,D_2)} + c_{(O_3,D_2)} - c_{(O_3,D_4)}$, ou simplesmente, $M(O_5, D_4) = -c_{52} + c_{32} - c_{34}$

Será demonstrado que após (p,q) entrar na base no lugar da variável (r_s, t_s), os novos custos atualizados $\bar{c}'_{ij}$ podem ser obtidos por meio de operações envolvendo somente $\bar{c}_{pq}$ e $\bar{c}_{ij}$, sem a necessidade do cálculo das variáveis duais u_i e v_j.

Antes do pivoteamento, tem-se a árvore $\mathcal{T}$, as subárvores $\mathcal{H}$ e $\mathcal{T} - \mathcal{H}$ e os caminhos de arcos $\Omega(s,t)$ entre os vértices $s,t \in \mathcal{K}$. Após o pivoteamento, tem-se a árvore $\mathcal{T}'$, as subárvores $\mathcal{H}'$ e $\mathcal{T} - \mathcal{H}$ e os caminhos de arcos $\Omega'(s,t)$ entre os vértices $s,t \in \mathcal{K}$.

A subárvore $\mathcal{T} - \mathcal{H}$ continua a mesma, já a árvore $\mathcal{H}$ deixa de ser enraizada em h^* e passa a ser enraizada em p ou q, para os casos em que $r_s \in \mathcal{A}(p)$ ou $t_s \in \mathcal{A}(q)$, respectivamente, conforme já explicado anteriormente e ilustrado nas figuras 30 a 34.

O custo atualizado de uma variável não básica associada ao arco (s,t), conforme já explicado no método do *Stepping Stone*, é obtido por meio de adições e subtrações, alternadamente, ao longo do ciclo formado pela inclusão de (s,t) na árvore $\mathcal{T}$ que representa a solução básica atual (ou no ciclo formado no quadro).

O ciclo formado pela inclusão de (p, q) em $\mathcal{T}$ pode ser representado também pela união de caminhos. Uma das possibilidades é considerar que o ciclo é a união dos seguintes caminhos:

- De p até q
- De q até o ancestral comum de p e q de maior profundidade, denotado por $\bar{\xi}$
- De $\bar{\xi}$ até p

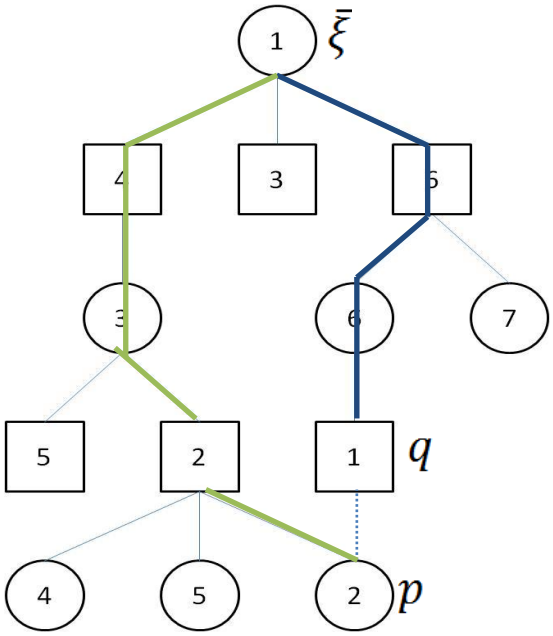


FIGURA 35 – EXEMPLO DE SUBCAMINHO NO $\theta - loop$
FONTE: O Autor (2014)

Para o decorrer das demonstrações a seguir, ressalta-se que o caminho $\Omega = P(s, t)$ de s até t numa árvore é único e que o ancestral comum de s e t de maior profundidade, pertence a $P(s, t)$.

A demonstração de quais são os novos custos atualizados $\bar{c}'_{ij}$ após o pivoteamento será realizada, no presente trabalho, para os quatro casos possíveis: o primeiro é o que $i \notin \mathcal{H}$ e $j \notin \mathcal{H}$, o segundo é o que $i \in \mathcal{H}$ e $j \in \mathcal{H}$, o terceiro é o que $i \notin \mathcal{H}$ e $j \in \mathcal{H}$ e, por fim, o quarto é o que $i \in \mathcal{H}$ e $j \notin \mathcal{H}$.

A fim de facilitar o entendimento da demonstração, são estabelecidos e demonstrados os lemas 1 a 4.

Lema 1: $\tilde{\alpha}M(\tilde{\xi},\bar{\xi}) = -\bar{\alpha}M(\bar{\xi},\hat{\xi})$

Demonstração: A demonstração é realizada em duas partes, primeiro mostra-se que a igualdade é válida para $\tilde{\alpha} = \bar{\alpha}$ e depois que também é verdadeira para $\tilde{\alpha} \neq \bar{\alpha}$.

1) Se $\tilde{\alpha} = \bar{\alpha}$

Neste caso, $\tilde{\xi},\bar{\xi} \in \mathcal{O}$ ou $\tilde{\xi},\bar{\xi} \in \mathcal{D}$. Desta forma, $|\Omega(\bar{\xi},\hat{\xi})|$ é par e, consequentemente, $M(\tilde{\xi},\bar{\xi}) = -M(\bar{\xi},\hat{\xi})$. Então, $\tilde{\alpha}M(\tilde{\xi},\bar{\xi}) = -\tilde{\alpha}M(\bar{\xi},\hat{\xi}) = -\bar{\alpha}M(\bar{\xi},\hat{\xi})$. Ou seja, $\tilde{\alpha}M(\tilde{\xi},\bar{\xi}) = -\bar{\alpha}M(\bar{\xi},\hat{\xi})$.

2) Se $\tilde{\alpha} \neq \bar{\alpha}$

Neste caso, $\tilde{\xi} \in \mathcal{O}$ e $\bar{\xi} \in \mathcal{D}$, ou $\bar{\xi} \in \mathcal{O}$ e $\tilde{\xi} \in \mathcal{D}$. Desta forma, $|\Omega(\bar{\xi},\tilde{\xi})|$ é ímpar e, consequentemente, $M(\tilde{\xi},\bar{\xi}) = M(\bar{\xi},\tilde{\xi})$. Então, $\tilde{\alpha}M(\tilde{\xi},\bar{\xi}) = \tilde{\alpha}M(\bar{\xi},\tilde{\xi}) = -\bar{\alpha}M(\bar{\xi},\tilde{\xi})$. Ou seja, $\tilde{\alpha}M(\tilde{\xi},\bar{\xi}) = -\bar{\alpha}M(\bar{\xi},\tilde{\xi})$.

Portanto, $\tilde{\alpha}M(\tilde{\xi},\bar{\xi}) = -\bar{\alpha}M(\bar{\xi},\tilde{\xi})$.

Lema 2: $\hat{\alpha}M(\hat{\xi},\bar{\xi}) = -\bar{\alpha}M(\bar{\xi},\hat{\xi})$ e $\hat{\alpha}M(\hat{\xi},\tilde{\xi}) = -\tilde{\alpha}M(\tilde{\xi},\hat{\xi})$. As demonstrações são análogas as apresentadas no lema 1 e, portanto, omitidas no presente trabalho.

Lema 3: $\tilde{\alpha}M(\tilde{\xi},p) = M(p,\tilde{\xi})$

Demonstração: A demonstração é dividida em duas partes: a primeira para quando $\tilde{\xi}$ é um vértice de destino, ou seja, $\tilde{\xi} \in \mathcal{D}$ e na sequência para quando $\tilde{\xi}$ é um vértice de origem, ou seja, $\tilde{\xi} \in \mathcal{O}$.

1) Se $\tilde{\xi} \in \mathcal{D}$

Neste caso, $\tilde{\alpha} = 1$ e $|\Omega(\tilde{\xi},p)|$ é ímpar e, consequentemente, $M(p,\tilde{\xi}) = M(\tilde{\xi},p)$. Logo $\tilde{\alpha}M(\tilde{\xi},p) = M(p,\tilde{\xi})$

2) Se $\tilde{\xi} \in \mathcal{O}$

Neste caso, $\tilde{\alpha} = -1$ e $|\Omega(\tilde{\xi},p)|$ é par e, consequentemente, $M(p,\tilde{\xi}) = -M(\tilde{\xi},p)$. Logo $\tilde{\alpha}M(\tilde{\xi},p) = M(p,\tilde{\xi})$

Portanto, $\tilde{\alpha}M(\tilde{\xi},p) = M(p,\tilde{\xi})$

Lema 4: $\bar{\alpha}M(\bar{\xi},q) = M(q,\bar{\xi})$

Demonstração: Assim como para o Lema 3, a demonstração é realizada em duas partes: a primeira para quando $\bar{\xi}$ é um vértice de destino, ou seja, $\bar{\xi} \in \mathcal{D}$ e na sequência para quando $\bar{\xi}$ é um vértice de origem, ou seja, $\bar{\xi} \in \mathcal{O}$

1) Se $\bar{\xi} \in \mathcal{D}$

Neste caso, $\bar{\alpha} = 1$ e $|\Omega(\bar{\xi},q)|$ é par e, consequentemente, $M(q,\bar{\xi}) = -M(\bar{\xi},q)$. Logo $\bar{\alpha}M(\bar{\xi},q) = -M(q,\bar{\xi})$

2) Se $\bar{\xi} \in \mathcal{O}$

Neste caso, $\bar{\alpha} = -1$ e $|\Omega(\bar{\xi},q)|$ é ímpar e, consequentemente, $M(q,\bar{\xi}) = M(\bar{\xi},q)$. Logo $\bar{\alpha}M(\bar{\xi},q) = -M(q,\bar{\xi})$

Portanto, $\bar{\alpha}M(\bar{\xi},q) = -M(q,\bar{\xi})$

Caso 1: $i \notin \mathcal{H}$ e $j \notin \mathcal{H}$

A subárvore $\mathcal{T} - \mathcal{H} \subset \mathcal{T}$ continua após o pivoteamento sendo uma subárvore de $\mathcal{T}'$, ou seja, $\mathcal{T} - \mathcal{H} \subset \mathcal{T}'$. Desta forma, a situação de que $i,j \notin \mathcal{H}$ é equivalente a $i,j \in \mathcal{T} - \mathcal{H}$. Assim, tem-se que $\bar{c}'_{ij} = \bar{c}_{ij}$, pois $\Omega'(i,j) = \Omega(i,j)$, isto é, o caminho do nó i ao nó j permanece o mesmo e, consequentemente, $\bar{c}'_{ij} = c_{ij} + M'(i,j) = c_{ij} + M(i,j) = \bar{c}_{ij}$.

Caso 2: $i \in \mathcal{H}$ e $j \in \mathcal{H}$

No processo de pivoteamento ocorre uma troca de raiz da subárvore $\mathcal{H} \subset \mathcal{T}$. A raiz de $\mathcal{H}$, h^*, passa a ser p ou q, mas o caminho entre dois vértices permanece o mesmo. Assim, de forma análoga ao já explicado para o Caso 1, tem-se que $\bar{c}'_{ij} = c_{ij} + M'(i,j) = c_{ij} + M(i,j) = \bar{c}_{ij}$.

Caso 3: $i \notin \mathcal{H}$ e $j \in \mathcal{H}$

Para este caso, faz-se necessário dividir a demonstração em duas partes, a primeira (Caso 3.1) para quando $r_s \in \mathcal{A}(p)$ e a segunda (Caso 3.2) quando $t_s \in \mathcal{A}(q)$.

Caso 3.1: $h^* = r_s$

Neste caso, a árvore $\mathcal{T}$ associada a solução básica factível atual $\mathcal{B}$ é apresentada na figura 36 (a), já a árvore $\mathcal{T}'$ associada a solução básica factível $\mathcal{B}'$ após a entrada de (p, q) no lugar de (r_s, t_s) é apresentada na figura 36 (b)

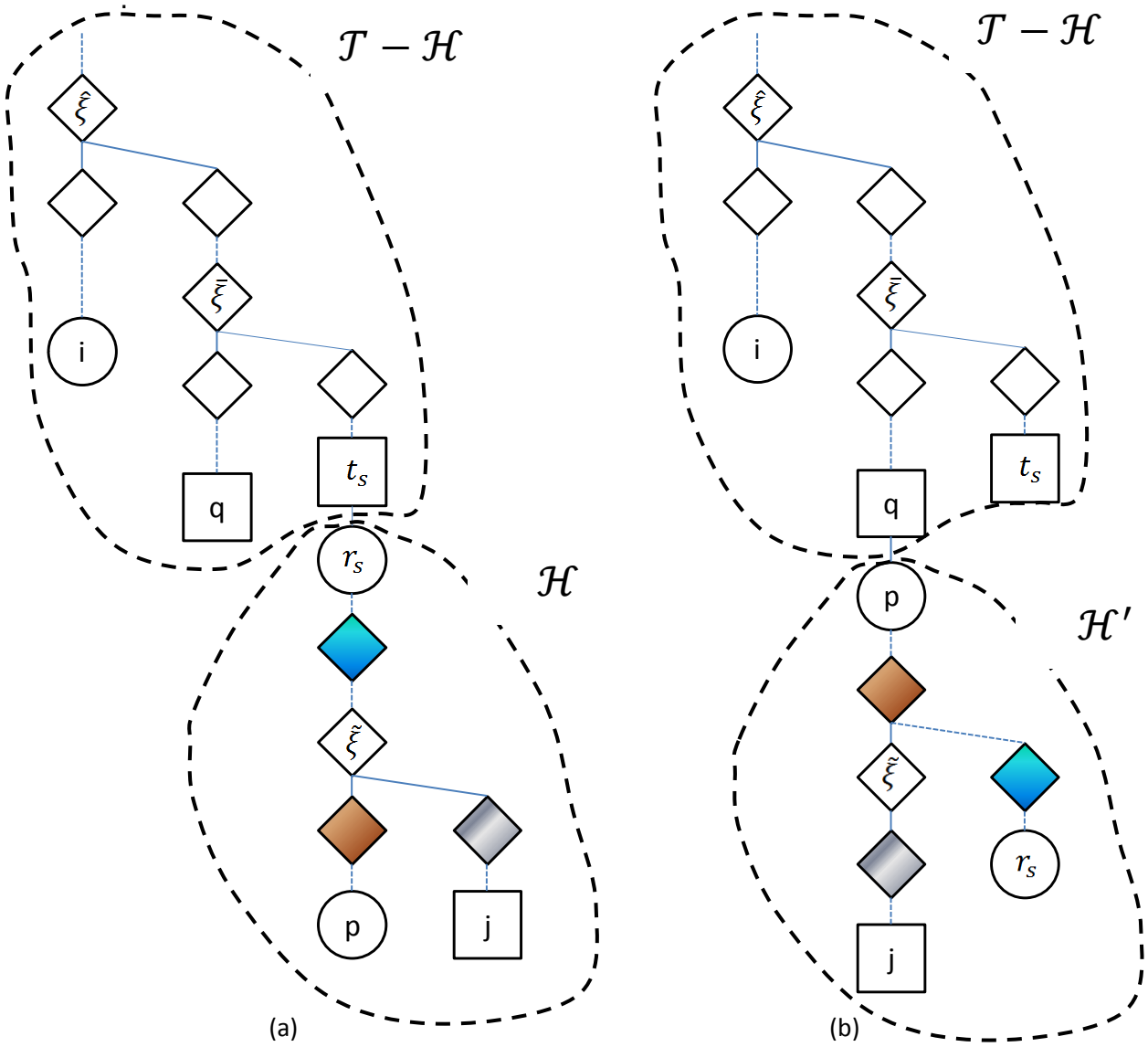


FIGURA 36 – REPRESENTAÇÃO DE $\mathcal{T}$ E $\mathcal{T}'$ PARA O CASO 3.1
FONTE: O Autor (2014)

Explicação sobre o valor de $\bar{c}_{pq}$: O caminho $\Omega = P(q,p)$ de q até p numa árvore é único. Como o vértice $\bar{\xi}$, ancestral comum de p e q de maior profundidade, pertence a Ω, tem-se que Ω pode ser reescrito como $\Omega = \Omega_1 \bar{\xi} \Omega_2$, onde $\Omega_1 = \Omega(q, \bar{\xi})$ e $\Omega_2 = \Omega(\bar{\xi}, p)$. Além disso, como $\tilde{\xi}$, ancestral comum de p e j de maior profundidade,

pertence a Ω_2, tem-se que Ω_2 pode ser reescrito como $\Omega_2 = \Omega_3\tilde{\xi}\Omega_4$, onde $\Omega_3 = \Omega(\bar{\xi}, \tilde{\xi})$ e $\Omega_4 = \Omega(\tilde{\xi}, p)$. Assim, $\Omega = \Omega_1\bar{\xi}\Omega_3\tilde{\xi}\Omega_4$ e $\bar{c}_{pq}$ é calculado pela expressão (26).

$$\bar{c}_{pq} = c_{pq} + M(q, \bar{\xi}) + \bar{\alpha}M(\bar{\xi}, \tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi}, p) \tag{26}$$

Explicação sobre o valor de $\bar{c}_{ij}$: O caminho $\Omega = P(j, i)$ de j até i numa árvore também é único. Como o vértice $\hat{\xi}$, ancestral comum de i e j de maior profundidade, pertence a Ω, tem-se que Ω pode ser reescrito como $\Omega = \Omega_1\hat{\xi}\Omega_2$, onde $\Omega_1 = \Omega(j, \hat{\xi})$ e $\Omega_2 = \Omega(\hat{\xi}, i)$. Além disso, $\tilde{\xi}$, ancestral comum de p e j de maior profundidade, pertence a Ω_1, logo Ω_1 pode ser reescrito como $\Omega_1 = \Omega_3\tilde{\xi}\Omega_4$, onde $\Omega_3 = \Omega(j, \tilde{\xi})$ e $\Omega_4 = \Omega(\tilde{\xi}, \hat{\xi})$. Assim, $\Omega = \Omega_3\tilde{\xi}\Omega_4\hat{\xi}\Omega_2$. Ainda tem-se que o vértice $\bar{\xi}$, ancestral comum de p e q de maior profundidade, pertence a Ω_4 e, consequentemente, Ω_4 pode ser reescrito como $\Omega_4 = \Omega_5\bar{\xi}\Omega_6$, onde $\Omega_5 = \Omega(\tilde{\xi}, \bar{\xi})$ e $\Omega_6 = \Omega(\bar{\xi}, \hat{\xi})$. Desta forma, por fim, $\Omega = \Omega_3\tilde{\xi}\Omega_5\bar{\xi}\Omega_6\hat{\xi}\Omega_2$ e $\bar{c}_{ij}$ é calculado pela expressão (27).

$$\bar{c}_{ij} = c_{ij} + M(j, \tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi}, \bar{\xi}) + \bar{\alpha}M(\bar{\xi}, \hat{\xi}) + \hat{\alpha}M(\hat{\xi}, i) \tag{27}$$

Explicação sobre o valor de $\bar{c}'_{ij}$: Na nova árvore $\mathcal{T}'$, o caminho $\Omega = P(j, i)$ de j até i também é único, mas é diferente do caminho em $\mathcal{T}$. Em $\mathcal{T}$, o caminho de j até i passava, na sequência, por $\tilde{\xi}$, $\bar{\xi}$ e $\hat{\xi}$. Já em $\mathcal{T}'$, o caminho passa, na sequência, por $\tilde{\xi}$, p, q, $\bar{\xi}$ e $\hat{\xi}$. A mudança de parte da sequência de $\tilde{\xi}$, $\bar{\xi}$ para $\tilde{\xi}$, p, q, $\bar{\xi}$ ocorre porque o arco $(r_s, t_s) \in \Omega(\tilde{\xi}, \bar{\xi})$, logo existe alteração em $\Omega(\tilde{\xi}, \bar{\xi})$ e, consequentemente, em $\Omega(j, \bar{\xi})$. Ao mesmo, o arco(p, q) foi adicionado a árvore e $(p, q) \in \Omega'(\tilde{\xi}, \bar{\xi})$. Assim, o novo caminho entre j e $\bar{\xi}$ passa a ser $\Omega'(j, \bar{\xi}) = \Omega(j, \tilde{\xi})\tilde{\xi}\Omega(\tilde{\xi}, p)p(p, q)q\Omega(q, \bar{\xi})$. Portanto,

$$\Omega'(j, i) = \Omega(j, \tilde{\xi})\tilde{\xi}\Omega(\tilde{\xi}, p)p(p, q)q\Omega(q, \bar{\xi})\bar{\xi}\Omega(\bar{\xi}, \hat{\xi})\hat{\xi}\Omega(\hat{\xi}, i) \tag{28}$$

Desta forma, $\bar{c}'_{ij}$ é calculado como pela expressão (29)

$$\bar{c}'_{ij} = c_{ij} + M(j,\tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi},p) + c_{pq} + M(q,\bar{\xi}) + \bar{\alpha}M(\bar{\xi},\hat{\xi}) + \hat{\alpha}M(\hat{\xi},i) \quad (29)$$

Adicionando $\tilde{\alpha}M(\tilde{\xi},\bar{\xi}) + \bar{\alpha}M(\bar{\xi},\tilde{\xi}) = 0$ (Lema 1) a expressão (29), obtém-se a expressão (30)

$$\bar{c}'_{ij} = c_{ij} + M(j,\tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi},\bar{\xi}) + \bar{\alpha}M(\bar{\xi},\tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi},p) + c_{pq} + M(q,\bar{\xi}) + \bar{\alpha}M(\bar{\xi},\hat{\xi}) + \hat{\alpha}M(\hat{\xi},i) \quad (30)$$

Reescrevendo os termos da expressão (30), obtém-se a expressão (31)

$$\bar{c}'_{ij} = c_{ij} + M(j,\tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi},\bar{\xi}) + \bar{\alpha}M(\bar{\xi},\hat{\xi}) + \hat{\alpha}M(\hat{\xi},i) + c_{pq} + M(q,\bar{\xi}) + \bar{\alpha}M(\bar{\xi},\tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi},p) \quad (31)$$

Portanto, com base nas expressões (31), (27) e (26), para $h^* = r_s$, $i \notin \mathcal{H}$ e $j \in \mathcal{H}$ tem-se a expressão (32).

$$\bar{c}'_{ij} = \bar{c}_{ij} + \bar{c}_{pq} \quad (32)$$

Caso 3.2: $h^* = t_s$

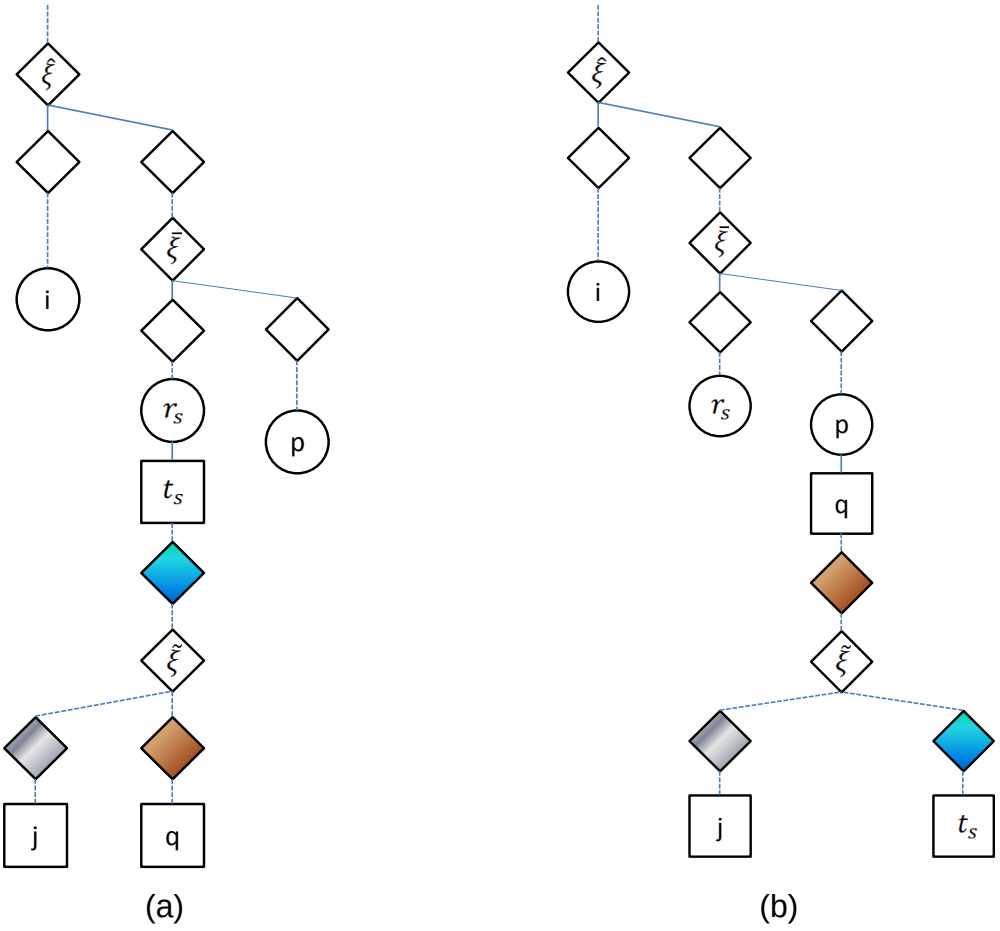


FIGURA 37 – REPRESENTAÇÃO DE $\mathcal{T}$ E $\mathcal{T}'$ PARA O CASO 3.2

FONTE: O Autor (2014)

Explicação sobre o valor de $\bar{c}_{pq}$: O caminho $\Omega = P(p,q)$ de p até q numa árvore é único. Como o vértice $\bar{\xi}$, ancestral comum de p e q de maior profundidade, pertence a Ω, tem-se que Ω pode ser reescrito como $\Omega = \Omega_1 \bar{\xi} \Omega_2$, onde $\Omega_1 = \Omega(q, \bar{\xi})$ e $\Omega_2 = \Omega(\bar{\xi}, p)$. Além disso, $\tilde{\xi}$, ancestral comum de q e j de maior profundidade, pertence a Ω_1, logo Ω_1 pode ser reescrito como $\Omega_1 = \Omega_3 \tilde{\xi} \Omega_4$, onde $\Omega_3 = \Omega(q, \tilde{\xi})$ e $\Omega_4 = \Omega(\tilde{\xi}, \bar{\xi})$. Assim, $\Omega = \Omega_3 \tilde{\xi} \Omega_4 \bar{\xi} \Omega_2$ e $\bar{c}_{pq}$ pode ser calculado pela expressão (33).

$$\bar{c}_{pq} = c_{pq} + M(q, \tilde{\xi}) + \tilde{\alpha} M(\tilde{\xi}, \bar{\xi}) + \bar{\alpha} M(\bar{\xi}, p) \qquad (33)$$

Explicação sobre o valor de $\bar{c}_{ij}$: O caminho $\Omega = P(j,i)$ de j até i numa árvore também é único. Como o vértice $\hat{\xi}$, ancestral comum de i e j de maior profundidade, pertence a Ω, tem-se que Ω pode ser reescrito como $\Omega = \Omega_1\hat{\xi}\Omega_2$, onde $\Omega_1 = \Omega(j,\hat{\xi})$ e $\Omega_2 = \Omega(\hat{\xi},i)$. Além disso, $\tilde{\xi}$, ancestral comum de q e j de maior profundidade, pertence a Ω_1, logo Ω_1 pode ser reescrito como $\Omega_1 = \Omega_3\tilde{\xi}\Omega_4$, onde $\Omega_3 = \Omega(j,\tilde{\xi})$ e $\Omega_4 = \Omega(\tilde{\xi},\hat{\xi})$. Assim, $\Omega = \Omega_3\tilde{\xi}\Omega_4\hat{\xi}\Omega_2$. Ainda tem-se que o vértice $\bar{\xi}$, ancestral comum de p e q de maior profundidade, pertence a Ω_4 e, consequentemente, Ω_4 pode ser reescrito como $\Omega_4 = \Omega_5\bar{\xi}\Omega_6$, onde $\Omega_5 = \Omega(\tilde{\xi},\bar{\xi})$ e $\Omega_6 = \Omega(\bar{\xi},\hat{\xi})$. Desta forma, por fim, $\Omega = \Omega_3\tilde{\xi}\Omega_5\bar{\xi}\Omega_6\hat{\xi}\Omega_2$ e $\bar{c}_{ij}$ pode ser calculado pela expressão (34).

$$\bar{c}_{ij} = c_{ij} + M(j,\tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi},\bar{\xi}) + \bar{\alpha}M(\bar{\xi},\hat{\xi}) + \hat{\alpha}M(\hat{\xi},i) \qquad (34)$$

Explicação sobre o valor de $\bar{c}'_{ij}$: Na nova árvore $\mathcal{T}'$, o caminho $\Omega = P(j,i)$ de j até i também é único, mas é diferente do caminho em $\mathcal{T}$. Em $\mathcal{T}$, o caminho de j até i passava, na sequência, por $\tilde{\xi}$, $\bar{\xi}$ e $\hat{\xi}$. Já em $\mathcal{T}'$, o caminho passa, na sequência, por $\tilde{\xi}$, q, p, $\bar{\xi}$ e $\hat{\xi}$. A mudança de parte da sequência de $\tilde{\xi}$, $\bar{\xi}$ para $\tilde{\xi}$, q, p, $\bar{\xi}$ ocorre porque o arco[18] $(r_s,t_s) \in \Omega(\tilde{\xi},\bar{\xi})$ deixa de fazer parte da árvore, logo existe alteração em $\Omega(\tilde{\xi},\bar{\xi})$ e, consequentemente, em $\Omega(j,\bar{\xi})$. Ao mesmo, o arco (p,q) foi adicionado a árvore e $(p,q) \in \Omega'(\tilde{\xi},\bar{\xi})$. Assim, o novo caminho entre j e $\bar{\xi}$ passa a ser $\Omega'(j,\bar{\xi}) = \Omega(j,\tilde{\xi})\tilde{\xi}\Omega(\tilde{\xi},q)q(p,q)p\Omega(p,\bar{\xi})$. Portanto,

$$\Omega'(j,i) = \Omega(j,\tilde{\xi})\tilde{\xi}\Omega(\tilde{\xi},q)q(p,q)p\Omega(p,\bar{\xi})\bar{\xi}\Omega(\bar{\xi},\hat{\xi})\hat{\xi}\Omega(\hat{\xi},i) \qquad (35)$$

Na expressão (36), tem-se $-c_{pq}$ porque o arco (p,q) está numa posição ímpar no caminho de arcos entre j e i. Desta forma, no $\theta - loop$ ele "aparece" com o sinal negativo.

$$\bar{c}'_{ij} = c_{ij} + M(j,\tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi},q) - c_{pq} - M(p,\bar{\xi}) + \bar{\alpha}M(\bar{\xi},\hat{\xi}) + \hat{\alpha}M(\hat{\xi},i) \qquad (36)$$

[18] Ressalta-se que $(r_s,t_s) \notin \Omega(q,\tilde{\xi})$, pois isso implicaria em $j \notin H$.

Para mostrar que, neste caso, $\bar{c}'_{ij} = \bar{c}_{ij} - \bar{c}_{pq}$, $\bar{c}_{pq}$ (33) será subtraído de $\bar{c}_{ij}$ (34) e o resultado obtido será $\bar{c}'_{ij}$ (36).

$$\bar{c}_{ij} - \bar{c}_{pq} = c_{ij} + M(j, \tilde{\xi}) + \tilde{\alpha} M(\tilde{\xi}, \bar{\xi}) + \bar{\alpha} M(\bar{\xi}, \hat{\xi}) + \hat{\alpha} M(\hat{\xi}, i) - c_{pq} - M(q, \tilde{\xi}) - \tilde{\alpha} M(\tilde{\xi}, \bar{\xi}) - \bar{\alpha} M(\bar{\xi}, p) \quad (37)$$

Eliminando $+\tilde{\alpha} M(\tilde{\xi}, \bar{\xi}) - \tilde{\alpha} M(\tilde{\xi}, \bar{\xi})$ e reescrevendo os termos de (37), tem-se a expressão (38)

$$\bar{c}_{ij} - \bar{c}_{pq} = c_{ij} + M(j, \tilde{\xi}) - M(q, \tilde{\xi}) - c_{pq} - \bar{\alpha} M(\bar{\xi}, p) + \bar{\alpha} M(\bar{\xi}, \hat{\xi}) + \hat{\alpha} M(\hat{\xi}, i) \quad (38)$$

Pelo Lema 4, $\tilde{\alpha} M(\tilde{\xi}, q) = -M(q, \tilde{\xi})$, logo a expressão (38) pode ser reescrita como sendo a expressão (39)

$$\bar{c}_{ij} - \bar{c}_{pq} = c_{ij} + M(j, \tilde{\xi}) + \tilde{\alpha} M(\tilde{\xi}, q) - c_{pq} - M(p, \bar{\xi}) + \bar{\alpha} M(\bar{\xi}, \hat{\xi}) + \hat{\alpha} M(\hat{\xi}, i) = \bar{c}'_{ij} \quad (39)$$

Portanto, $\bar{c}'_{ij} = \bar{c}_{ij} - \bar{c}_{pq}$ para $h^* = t_s$, $i \notin \mathcal{H}$ e $j \in \mathcal{H}$.

Caso 4: $i \in \mathcal{H}$ e $j \notin \mathcal{H}$

Para este caso, assim como para o caso 3, faz-se necessário dividir a demonstração em duas partes, a primeira (Caso 4.1) para quando $t_s \in \mathcal{A}(q)$ e a segunda (Caso 4.2) quando $r_s \in \mathcal{A}(p)$.

Caso 4.1: $h^* = t_s$

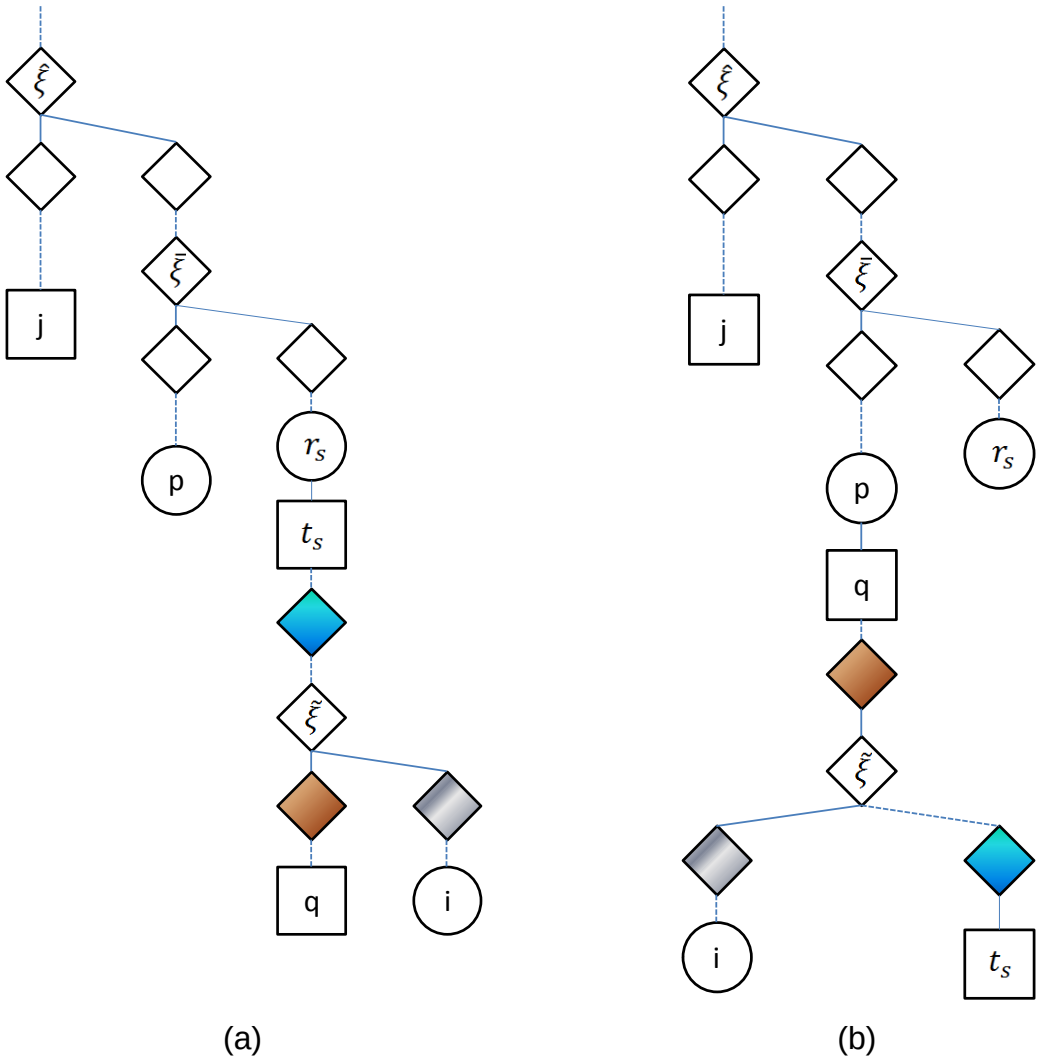


FIGURA 38 – REPRESENTAÇÃO DE $\mathcal{T}$ E $\mathcal{T}'$ PARA O CASO 4.1
FONTE: O Autor (2014)

Explicação sobre o valor de $\bar{c}_{ij}$: O caminho $\Omega = P(j,i)$ de j até i numa árvore também é único. Como o vértice $\hat{\xi}$, ancestral comum de i e j de maior profundidade, pertence a Ω, tem-se que Ω pode ser reescrito como $\Omega = \Omega_1\hat{\xi}\Omega_2$, onde $\Omega_1 = \Omega(j,\hat{\xi})$ e $\Omega_2 = \Omega(\hat{\xi},i)$. Além disso, $\tilde{\xi}$, ancestral comum de i e q de maior profundidade, pertence a Ω_2, logo Ω_2 pode ser reescrito como $\Omega_2 = \Omega_3\tilde{\xi}\Omega_4$, onde $\Omega_3 = \Omega(\hat{\xi},\tilde{\xi})$ e $\Omega_4 = \Omega(\tilde{\xi},i)$. Assim, $\Omega = \Omega_1\hat{\xi}\Omega_3\tilde{\xi}\Omega_4$. Ainda tem-se que o vértice $\bar{\xi}$, ancestral comum de p e q de maior profundidade, pertence a Ω_3 e, consequentemente, Ω_3 pode ser reescrito como $\Omega_3 = \Omega_5\bar{\xi}\Omega_6$, onde $\Omega_5 = \Omega(\hat{\xi},\bar{\xi})$ e

$\Omega_6 = \Omega(\bar{\xi}, \tilde{\xi})$. Desta forma, por fim, $\Omega = \Omega_1 \hat{\xi} \Omega_5 \bar{\xi} \Omega_6 \tilde{\xi} \Omega_4$ e $\bar{c}_{ij}$ pode ser calculado pela expressão (40)

$$\bar{c}_{ij} = c_{ij} + M(j, \hat{\xi}) + \hat{\alpha} M(\hat{\xi}, \bar{\xi}) + \bar{\alpha} M(\bar{\xi}, \tilde{\xi}) + \tilde{\alpha} M(\tilde{\xi}, i) \quad (40)$$

Explicação sobre o valor de $\bar{c}_{pq}$: O caminho $\Omega = P(q,p)$ de q até p numa árvore é único. Como o vértice $\bar{\xi}$, ancestral comum de p e q de maior profundidade, pertence a Ω, tem-se que Ω pode ser reescrito como $\Omega = \Omega_1 \bar{\xi} \Omega_2$, onde $\Omega_1 = \Omega(q, \bar{\xi})$ e $\Omega_2 = \Omega(\bar{\xi}, p)$. Além disso, $\tilde{\xi}$, ancestral comum de q e i de maior profundidade, pertence a Ω_1, tem-se que Ω_1 pode ser reescrito como $\Omega_1 = \Omega_3 \tilde{\xi} \Omega_4$, onde $\Omega_3 = \Omega(q, \tilde{\xi})$ e $\Omega_4 = \Omega(\tilde{\xi}, \bar{\xi})$. Assim, $\Omega = \Omega_3 \tilde{\xi} \Omega_4 \bar{\xi} \Omega_2$ e $\bar{c}_{pq}$ pode ser calculado pela expressão (41)

$$\bar{c}_{pq} = c_{pq} + M(q, \tilde{\xi}) + \tilde{\alpha} M(\tilde{\xi}, \bar{\xi}) + \bar{\alpha} M(\bar{\xi}, p) \quad (41)$$

Explicação sobre o valor de $\bar{c}'_{ij}$: Na nova árvore $\mathcal{T}'$, o caminho $\Omega = P(j,i)$ de j até i também é único, mas é diferente do caminho em $\mathcal{T}$. Em $\mathcal{T}$, o caminho de j até i passava, na sequência, por $\hat{\xi}$, $\bar{\xi}$ e $\tilde{\xi}$. Já em $\mathcal{T}'$, o caminho passa, na sequência, por $\hat{\xi}$, $\bar{\xi}$, p, q e $\tilde{\xi}$. A mudança de parte da sequência de $\bar{\xi}$, $\tilde{\xi}$ para $\bar{\xi}$, p, q, $\tilde{\xi}$ ocorre porque o arco[19] $(r_s, t_s) \in \Omega(\bar{\xi}, \tilde{\xi})$ deixa de fazer parte da árvore, logo existe alteração em $\Omega(\bar{\xi}, \tilde{\xi})$ e, consequentemente, em $\Omega(j, \tilde{\xi})$. Ao mesmo, o arco (p,q) foi adicionado a árvore e $(p,q) \in \Omega'(\bar{\xi}, \tilde{\xi})$. Assim, o novo caminho entre j e $\tilde{\xi}$ passa a ser $\Omega'(j, \tilde{\xi}) = \Omega(j, \hat{\xi}) \hat{\xi} \Omega(\hat{\xi}, \bar{\xi}) \bar{\xi} \Omega(\bar{\xi}, p) p(p,q) q \Omega(q, \tilde{\xi})$. Portanto, $\bar{c}'_{ij}$ pode ser calculado pela expressão (42)

$$\bar{c}'_{ij} = c_{ij} + M(j, \hat{\xi}) + \hat{\alpha} M(\hat{\xi}, \bar{\xi}) + \bar{\alpha} M(\bar{\xi}, p) + c_{pq} + M(q, \tilde{\xi}) + \tilde{\alpha} M(\tilde{\xi}, i) \quad (42)$$

Calculando $\bar{c}_{ij} + \bar{c}_{pq}$, representados pelas expressões (40) e (41), respectivamente, tem-se a expressão (43)

[19] Ressalta-se que $(r_s, t_s) \notin \Omega(q, \tilde{\xi})$, pois isso implicaria em $j \notin H$.

$$\bar{c}_{ij} + \bar{c}_{pq} = c_{ij} + M(j,\hat{\xi}) + \hat{\alpha}M(\hat{\xi},\bar{\xi}) + \bar{\alpha}M(\bar{\xi},p) + c_{pq} + M(q,\tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi},i) + \tilde{\alpha}M(\tilde{\xi},\bar{\xi}) + \bar{\alpha}M(\bar{\xi},\tilde{\xi}) \quad (43)$$

Como, pelo Lema 1, $\tilde{\alpha}M(\tilde{\xi},\bar{\xi}) = -\bar{\alpha}M(\bar{\xi},\tilde{\xi})$, tem-se

$$\bar{c}_{ij} + \bar{c}_{pq} = c_{ij} + M(j,\hat{\xi}) + \hat{\alpha}M(\hat{\xi},\bar{\xi}) + \bar{\alpha}M(\bar{\xi},p) + c_{pq} + M(q,\tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi},i) = \bar{c}'_{ij} \quad (44)$$

$$\therefore \bar{c}'_{ij} = \bar{c}_{ij} + \bar{c}_{pq}, i \in \mathcal{H}, j \notin \mathcal{H}, h^* \in \mathcal{A}(q) \quad (45)$$

Caso 4.2: $h^* = r_s$

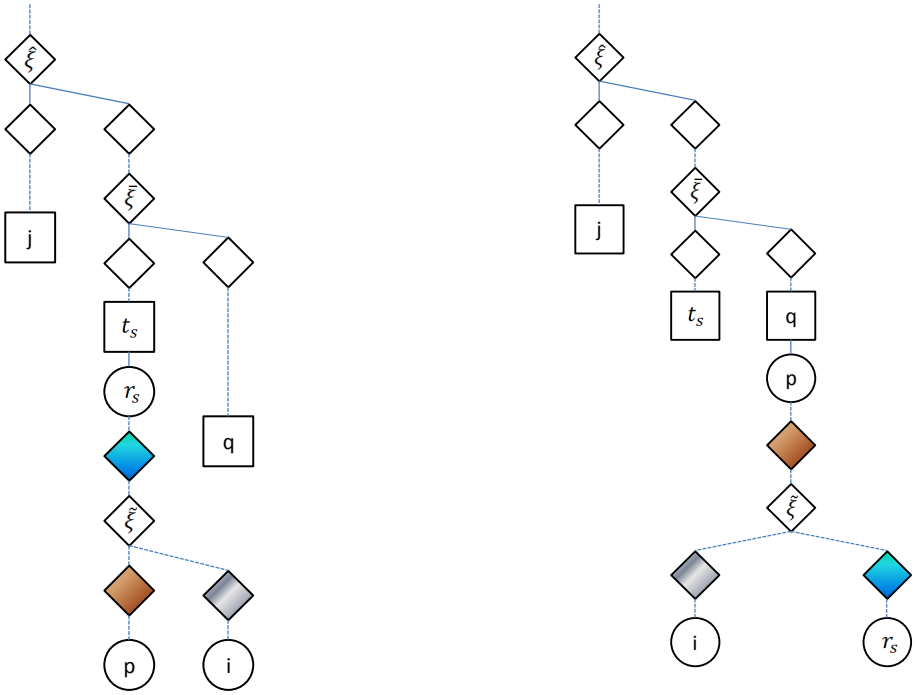


FIGURA 39 – REPRESENTAÇÃO DE $\mathcal{T}$ E $\mathcal{T}'$ PARA O CASO 4.2
FONTE: O Autor (2014)

Explicação sobre o valor de $\bar{c}_{ij}$: O caminho $\Omega = P(j,i)$ de j até i numa árvore também é único. Como o vértice $\hat{\xi}$, ancestral comum de i e j de maior profundidade, pertence a Ω, tem-se que Ω pode ser reescrito como $\Omega = \Omega_1\hat{\xi}\Omega_2$, onde $\Omega_1 = \Omega(j,\hat{\xi})$ e $\Omega_2 = \Omega(\hat{\xi},i)$. Além disso, $\tilde{\xi}$, ancestral comum de i e p de maior

profundidade, pertence a Ω_2, logo Ω_2 pode ser reescrito como $\Omega_2 = \Omega_3\tilde{\xi}\Omega_4$, onde $\Omega_3 = \Omega(\hat{\xi}, \tilde{\xi})$ e $\Omega_4 = \Omega(\tilde{\xi}, i)$. Assim, $\Omega = \Omega_1\hat{\xi}\Omega_3\tilde{\xi}\Omega_4$. Ainda tem-se que o vértice $\bar{\xi}$, ancestral comum de p e q de maior profundidade, pertence a Ω_3 e, consequentemente, Ω_3 pode ser reescrito como $\Omega_3 = \Omega_5\bar{\xi}\Omega_6$, onde $\Omega_5 = \Omega(\hat{\xi}, \bar{\xi})$ e $\Omega_6 = \Omega(\bar{\xi}, \tilde{\xi})$. Desta forma, por fim, $\Omega = \Omega_1\hat{\xi}\Omega_5\bar{\xi}\Omega_6\tilde{\xi}\Omega_4$ e $\bar{c}_{ij}$ pode ser calculado pela expressão (46).

$$\bar{c}_{ij} = c_{ij} + M(j, \hat{\xi}) + \hat{\alpha}M(\hat{\xi}, \bar{\xi}) + \bar{\alpha}M(\bar{\xi}, \tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi}, i) \quad (46)$$

Explicação sobre o valor de $\bar{c}_{pq}$: O caminho $\Omega = P(q, p)$ de q até p numa árvore é único. Como o vértice $\bar{\xi}$, ancestral comum de p e q de maior profundidade, pertence a Ω, tem-se que Ω pode ser reescrito como $\Omega = \Omega_1\bar{\xi}\Omega_2$, onde $\Omega_1 = \Omega(q, \bar{\xi})$ e $\Omega_2 = \Omega(\bar{\xi}, p)$. Além disso, como $\tilde{\xi}$, ancestral comum de p e i de maior profundidade, pertence a Ω_2, tem-se que Ω_2 pode ser reescrito como $\Omega_2 = \Omega_3\tilde{\xi}\Omega_4$, onde $\Omega_3 = \Omega(\bar{\xi}, \tilde{\xi})$ e $\Omega_4 = \Omega(\tilde{\xi}, p)$. Assim, $\Omega = \Omega_1\bar{\xi}\Omega_3\tilde{\xi}\Omega_4$ e $\bar{c}_{pq}$ pode ser calculado pela expressão (47)

$$\bar{c}_{pq} = c_{pq} + M(q, \bar{\xi}) + \bar{\alpha}M(\bar{\xi}, \tilde{\xi}) + \tilde{\alpha}M(\tilde{\xi}, p) \quad (47)$$

Explicação sobre o valor de $\bar{c}'_{ij}$: Na nova árvore $\mathcal{T}'$, o caminho $\Omega = P(j, i)$ de j até i também é único, mas é diferente do caminho em $\mathcal{T}$. Em $\mathcal{T}$, o caminho de j até i passava, na sequência, por $\hat{\xi}$, $\bar{\xi}$ e $\tilde{\xi}$. Já em $\mathcal{T}'$, o caminho passa, na sequência, por $\hat{\xi}$, $\bar{\xi}$, q, p e $\tilde{\xi}$. A mudança de parte da sequência de $\bar{\xi}$, $\tilde{\xi}$ para $\bar{\xi}$, q, p, $\tilde{\xi}$ ocorre porque o arco[20] $(r_s, t_s) \in \Omega(\bar{\xi}, \tilde{\xi})$ deixa de fazer parte da árvore, logo existe alteração em $\Omega(\bar{\xi}, \tilde{\xi})$ e, consequentemente, em $\Omega(j, \tilde{\xi})$. Ao mesmo, o arco (p, q) foi adicionado a árvore e $(p, q) \in \Omega'(\bar{\xi}, \tilde{\xi})$. Assim, o novo caminho entre j e $\tilde{\xi}$ passa a ser $\Omega'(j, \tilde{\xi}) = \Omega(j, \hat{\xi})\hat{\xi}\Omega(\hat{\xi}, \bar{\xi})\bar{\xi}\Omega(\bar{\xi}, q)q(p, q)p\Omega(p, \tilde{\xi})$. Portanto, $\bar{c}'_{ij}$ pode ser calculado pela expressão (48)

[20] Ressalta-se que $(r_s, t_s) \notin \Omega(p, \tilde{\xi})$, pois isso implicaria em $i \notin \mathcal{H}$.

$$\bar{c}'_{ij} = c_{ij} + M\left(j, \hat{\xi}\right) + \hat{\alpha}M\left(\hat{\xi}, \bar{\xi}\right) + \bar{\alpha}M\left(\bar{\xi}, q\right) - c_{pq} - M\left(p, \tilde{\xi}\right) + \tilde{\alpha}M\left(\tilde{\xi}, i\right) \quad (48)$$

Calculando $\bar{c}_{ij} - \bar{c}_{pq}$, representados pelas expressões (46) e (47), respectivamente, obtém-se a expressão (49)

$$\begin{aligned}\bar{c}_{ij} - \bar{c}_{pq} = c_{ij} + M\left(j, \hat{\xi}\right) + \hat{\alpha}M\left(\hat{\xi}, \bar{\xi}\right) + \bar{\alpha}M\left(\bar{\xi}, \tilde{\xi}\right) + \tilde{\alpha}M\left(\tilde{\xi}, i\right) - c_{pq} - M\left(q, \bar{\xi}\right) \\ - \bar{\alpha}M\left(\bar{\xi}, \tilde{\xi}\right) - \tilde{\alpha}M\left(\tilde{\xi}, p\right)\end{aligned} \quad (49)$$

Reescrevendo a expressão (49), tem-se a expressão (50)

$$\begin{aligned}\bar{c}_{ij} - \bar{c}_{pq} = c_{ij} + M\left(j, \hat{\xi}\right) + \hat{\alpha}M\left(\hat{\xi}, \bar{\xi}\right) - M\left(q, \bar{\xi}\right) - c_{pq} - \tilde{\alpha}M\left(\tilde{\xi}, p\right) + \tilde{\alpha}M\left(\tilde{\xi}, i\right) \\ + \bar{\alpha}M\left(\bar{\xi}, \tilde{\xi}\right) - \bar{\alpha}M\left(\bar{\xi}, \tilde{\xi}\right)\end{aligned} \quad (50)$$

A partir da expressão (50), tem-se a expressão (51)

$$\bar{c}_{ij} - \bar{c}_{pq} = c_{ij} + M\left(j, \hat{\xi}\right) + \hat{\alpha}M\left(\hat{\xi}, \bar{\xi}\right) - M\left(q, \bar{\xi}\right) - c_{pq} - \tilde{\alpha}M\left(\tilde{\xi}, p\right) + \tilde{\alpha}M\left(\tilde{\xi}, i\right) \quad (51)$$

Como $M\left(q, \bar{\xi}\right) = -\bar{\alpha}M\left(\bar{\xi}, q\right)$ e $\tilde{\alpha}M\left(\tilde{\xi}, p\right) = M\left(p, \tilde{\xi}\right)$, obtém-se de (48) e (51) a expressão (52)

$$\begin{aligned}\bar{c}_{ij} - \bar{c}_{pq} = c_{ij} + M\left(j, \hat{\xi}\right) + \hat{\alpha}M\left(\hat{\xi}, \bar{\xi}\right) + \bar{\alpha}M\left(\bar{\xi}, q\right) - c_{pq} - M\left(p, \tilde{\xi}\right) + \tilde{\alpha}M\left(\tilde{\xi}, i\right) \\ = \bar{c}'_{ij}\end{aligned} \quad (52)$$

$$\therefore \bar{c}'_{ij} = \bar{c}_{ij} - \bar{c}_{pq}, i \in \mathcal{H}, j \notin \mathcal{H}, h^* \epsilon \mathcal{A}(p) \quad (53)$$

Desta forma, conclui-se que os novos valores de custos atualizados para as variáveis não básicas, após a variável x_{pq} entrar na base no lugar da variável x_{r_s,t_s} podem ser obtidos pela expressão (54)

$$\bar{c}'_{ij} = \begin{cases} \bar{c}_{ij}, se\ i,j \notin \mathcal{H}\ ou\ i,j \in \mathcal{H} \\ \bar{c}_{ij} + \bar{c}_{pq}, se\ i \notin \mathcal{H}\ e\ j \in \mathcal{H}, r_s \epsilon \mathcal{A}(p) \\ \bar{c}_{ij} + \bar{c}_{pq}, se\ i \in \mathcal{H}, j \notin \mathcal{H}, t_s \epsilon \mathcal{A}(q) \\ \bar{c}_{ij} - \bar{c}_{pq}, se\ i \notin \mathcal{H}\ e\ j \in \mathcal{H}, t_s \epsilon \mathcal{A}(q) \\ \bar{c}_{ij} - \bar{c}_{pq}, se\ i \in \mathcal{H}, j \notin \mathcal{H}, r_s \epsilon \mathcal{A}(p) \end{cases} \quad (54)$$

Na busca por um algoritmo capaz de resolver um problema de programação matemática em baixo tempo computacional são igualmente importantes os resultados matemáticos teóricos, como o apresentado neste capítulo, a estrutura de dados utilizadas e a forma como os valores das informações são atualizadas na estrutura escolhida.

A diferença de tempo de resolução do PT entre se calcular os valores das variáveis duais e todos os custos atualizados quando comparada com a situação de recalcular somente os custos atualizados necessários conforme expressão (54) dependerá da significância, em termos de operações e tempo, desta parte do algoritmo perante o programa completo. Por exemplo, se o recálculo dos custos atualizados pela forma do método MODI for de 10% do tempo total de resolução do problema, uma redução de 70% neste procedimento gera um ganho de apenas 7% no tempo total do algoritmo, já se o recálculo dos custos atualizados pela forma do método MODI for de 80% do tempo total de resolução do problema, uma mesma redução de 70% neste procedimento gera um ganho de 56% no tempo total do algoritmo.

No próximo capítulo é descrita a implementação computacional utilizada para os resultados apresentados no capítulo 5, onde foram analisados ganhos realizados pela utilização de diferentes estruturas, do resultado matemático do presente capítulo e também de testes adicionais para buscar um melhor tempo computacional ainda.

4 IMPLEMENTAÇÃO COMPUTACIONAL

A implementação computacional dos algoritmos para a resolução do PT foi realizada na linguagem VB.NET. Neste capítulo apresenta-se as estruturas utilizadas para armazenamento dos dados, o pseudocódigo e também um exemplo numérico para facilitar o entendimento e possibilitar que esta implementação seja reproduzida por outros interessados.

4.1 AS ESTRUTURAS DE DADOS UTILIZADAS

Como citado por FOX (1978), a estrutura de dados utilizada desempenha papel importante nas implementações de rotinas em Pesquisa Operacional. Desta forma, o tempo de execução de um algoritmo não depende apenas da sua ordem de complexidade matemática teórica, mas também da maneira como as informações são armazenadas e atualizadas.

Durante o desenvolvimento da presente tese, diversas estruturas de dados foram testadas, adaptadas e melhoradas. Por questões de objetividade, será apresentada aqui apenas a estrutura de dados que permitiu os melhores resultados em tempos computacionais.

4.1.1 Estrutura de dados para representação do Problema de Transporte

Um Problema de Transporte pode ser definido pela matriz de custos, vetor de ofertas e vetor de demandas. Diante disso, a estrutura para armazenar o problema deve conter, no mínimo, estas informações. Como o interesse é a implementação de um método para resolução do problema, na mesma estrutura que armazena os dados relativos a formulação do problema, também são armazenadas as informações referentes à solução primal e às variáveis duais do problema.

```
Estrutura strTransporte
    MatrizCustoVariavel(,)
    CustoTotal
    VetorOfertas()
    VetorDemandas()
    SolucaoInicial(,)
    VetorU()
    VetorV()
Fim Estrutura
```

QUADRO 9 – ESTRUTUTA **strTransporte** PARA ARMAZENAMENTO DAS INFORMAÇÕES DE UM PROBLEMA DE TRANSPORTE

FONTE: O Autor (2014)

Com uma variável do tipo **strTransporte**, é possível armazenar todas as informações de um problema de transporte: nas variáveis *MatrizCustoVariável(,)*, *VetorOfertas()* e *VetorDemandas()* são armazenadas, respectivamente, a matriz C da expressão (1), vetor a da expressão (2) e vetor b da expressão (3); na variável *SoluçãoInicial(,)* é armazenada em forma de lista a solução atual; nas variáveis *VetorU()* e *VetorV()* são armazenados os valores das variáveis duais; na variável *CustoTotal* é armazenado o valor da função objetivo.

Um exemplo para facilitar o entendimento sobre o armazenamento de informações nesta estrutura é a variável chamada ***ProblemaTransporte***, do tipo **strTransporte**, apresentada na figura 40. Neste exemplo o valor de ***ProblemaTransporte***.*MatrizCustoVariavel(2,5)* é 4, já o valor de ***ProblemaTransporte***.*CustoTotal* é 703.

ProblemaTransporte	*MatrizCustoVariavel*	$\begin{bmatrix} 7 & 3 & 4 & 4 & 7 \\ 5 & 5 & 5 & 7 & 4 \\ 3 & 6 & 3 & 6 & 5 \\ 3 & 4 & 7 & 7 & 5 \\ 7 & 7 & 4 & 7 & 5 \end{bmatrix}$
	CustoTotal	703
	VetorOfertas	$[22 \quad 44 \quad 34 \quad 7 \quad 44]$
	VetorDemandas	$[21 \quad 35 \quad 6 \quad 47 \quad 42]$

	SolucaoInicial	$\begin{bmatrix} 22 & 1 & 2 \\ 21 & 3 & 1 \\ 6 & 3 & 3 \\ 42 & 2 & 5 \\ 7 & 4 & 2 \\ 2 & 2 & 2 \\ 4 & 3 & 2 \\ 3 & 3 & 4 \\ 44 & 5 & 4 \end{bmatrix}$
	VetorU	$[0 \quad 2 \quad 3 \quad 1 \quad 4]$
	VetorV	$[0 \quad 3 \quad 0 \quad 3 \quad 2]$

FIGURA 40 – EXEMPLO DE VALORES PARA A VARIÁVEL ***ProblemaTransporte*** DO TIPO **strTransporte**

FONTE: O Autor (2014)

Ressalta-se que na figura 40 os valores de ***ProblemaTransporte****.VetorU()* e ***ProblemaTransporte****.VetorU()* não estarão preenchidos inicialmente por não serem aqui consideradas informações iniciais do problema, mas para ilustração do exemplo aqui eles foram completados.

4.1.2 Estrutura de dados para representação das variáveis básicas e não básicas

Para que seja possível uma implementação que permita a resolução do problema num menor tempo computacional, também é necessário que informações estejam armazenadas de forma conveniente para que não sejam necessárias a realização de procuras excessivas. Em algumas situações isso gerou algumas duplicidades de informação na implementação, mas o tempo de resolução foi menor.

Como já explicado anteriormente, um PT com m origens e n destinos, possui $m + n - 1$ variáveis básicas e $m.n - (m + n - 1)$ variáveis não básicas. Estes dois tipos de variáveis possuem características diferentes. Por exemplo, para as variáveis básicas o valor da variável pode ser diferente de zero, enquanto que as variáveis não básicas podem possuir, e geralmente possuem, valores custo atualizado diferentes de zero. Desta forma, foram criadas estruturas separadas para armazenar informações referentes às variáveis básicas (quadro 10) e referentes às variáveis não básicas (quadro 11).

```
Estrutura strBasicas
    Linha
    Coluna
    Quantidade
    CustoVariavel
    CustoTotal
Fim Estrutura
```

QUADRO 10 – ESTRUTURA **strBasicas** PARA ARMAZENAMENTO DAS INFORMAÇÕES DAS VARIÁVEIS BÁSICAS DA SOLUÇÃO ATUAL NO PROBLEMA DE TRANSPORTE

FONTE: O Autor (2014)

Um exemplo do armazenamento de informações numa variável chamada ***Basicas()***, do tipo **strBasicas**, é apresentado na figura 41 utilizando como referência os valores da variável ***ProblemaTransporte*** da figura 40.

Basicas()					
posição	*Linha*	*Coluna*	*Quantidade*	*CustoVariavel*	*CustoTotal*
1	1	2	22	3	66
2	3	1	21	3	63
3	3	3	6	3	18
4	2	5	42	4	168
5	4	2	7	4	28
6	2	2	2	5	10
7	3	2	4	6	24
8	3	4	3	6	18
9	5	4	44	7	308

FIGURA 41 – EXEMPLO DE UMA VARIÁVEL DO TIPO **srtBasicas** NO PROBLEMA DE TRANSPORTE

FONTE: O Autor (2014)

Como exemplo de registro de informações da variável ***Basicas()*** apresentada na figura 41, tem-se que na posição 4 a variável *Linha* vale 2, a variável *Coluna* vale 5, a variável *Quantidade* vale 42, a variável *CustoVariável* 4 e *CustoTotal* 168, ou seja, tem-se que ***Basicas(4).****Linha*=2, ***Basicas(4).****CustoVariavel*=4 e ***Basicas(4).****CustoTotal*=168.

```
Estrutura strNaoBasicas
    Linha
    Coluna
    CustoVariavel
    CustoAtualizado
Fim Estrutura
```

QUADRO 11 – ESTRUTURA **strNaoBasicas** PARA ARMAZENAMENTO DAS INFORMAÇÕES DAS VARIÁVEIS BÁSICAS DA SOLUÇÃO ATUAL NO PROBLEMA DE TRANSPORTE

FONTE: O Autor (2014)

Um exemplo do armazenamento de informações numa variável chamada ***NaoBasicas()***, do tipo **strNaoBasicas**, é apresentado na figura 42 utilizando como referência os valores da variável ***ProblemaTransporte*** da figura 40.

NaoBasicas()				
posição	*Linha*	*Coluna*	*CustoVariavel*	*CustoAtualizado*
1	1	1	7	7
2	1	3	4	4
3	1	4	4	1
4	1	5	7	5
5	2	1	5	3
6	2	3	5	3
7	2	4	7	2
8	3	5	5	0
9	4	1	3	2
10	4	3	7	6
11	4	4	7	3
12	4	5	5	2
13	5	1	7	3
14	5	2	7	0
15	5	3	4	0
16	5	5	5	-1

FIGURA 42 – EXEMPLO DE UMA VARIÁVEL DO TIPO **srtNaoBasicas** NO PROBLEMA DE TRANSPORTE

FONTE: O Autor (2014)

Como exemplo de registro de informações da variável ***NaoBasicas()*** apresentada na figura 42, tem-se que na posição 14 a variável *Linha* vale 5, a variável *Coluna* vale 2, a variável *CustoVariável* tem o valor 7 e *CustoAtualizado* vale 0, ou seja, tem-se que ***NaoBasicas*(**14**).***Linha*=5, ***NaoBasicas*(**14**).***Coluna*=2, ***NaoBasicas*(**14**).***CustoVariavel*=7 e ***NaoBasicas*(**14**).***CustoAtualizado*=0.

4.1.3 Estrutura de dados para representação da árvore geradora

Uma árvore geradora que representa uma solução básica factível para o PT possui um nó raiz que pode ser qualquer origem ou destino. Sem perda de generalidade, será escolhido o vértice $\mathcal{O}_1$ para ser o nó raiz[21] em todos os exemplos deste capítulo. Cada um dos nós possui um nó pai e pode possuir, ou não, nós filhos.

Para a solução básica factível apresentada na figura 40, a representação da árvore geradora associada, considerando o vértice $\mathcal{O}_1$ como raiz, é apresentada na figura 43.

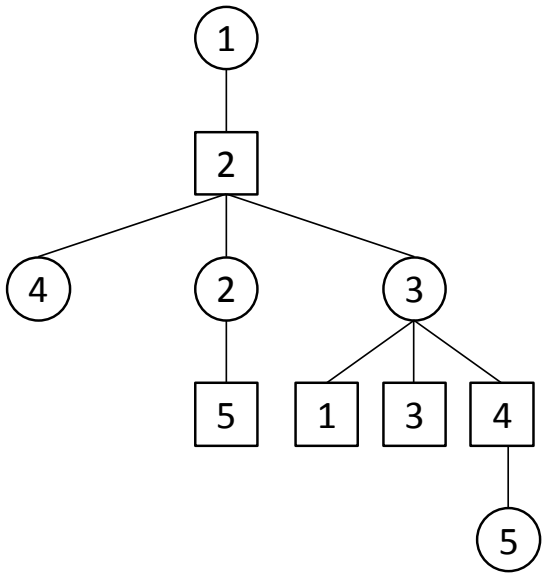


FIGURA 43 – REPRESENTAÇÃO DA ÁRVORE GERADORA PARA A SOLUÇÃO BÁSICA FACTÍVEL DA FIGURA 40

FONTE: O Autor (2014)

[21] Nos testes computacionais realizados, sempre a raiz da árvore geradora foi o vértice $\mathcal{O}_1$

Cada nó, com exceção da raiz, na árvore geradora possui um nó pai e, inclusive a raiz, uma lista de filhos. Desta forma, para armazenar as informações que definem a árvore geradora foi criada a estrutura **strNoArvore** (figura 44).

```
Estrutura strNoArvore
    ListaDeFilhos
    Pai
Fim Estrutura
```

FIGURA 44 – ESTRUTURA **strNoArvore** PARA ARMAZENAMENTO DAS INFORMAÇÕES DAS VARIÁVEIS DA ÁRVORE GERADORA REFERENTE A SOLUÇÃO BÁSICA ATUAL NO PROBLEMA DE TRANSPORTE

FONTE: O Autor (2014)

Embora fosse possível dimensionar uma variável, chamada nó, por exemplo, como sendo um vetor do tipo **strNoArvore** de $m + n$ posições, onde da posição 1 até a posição m seria referente aos nós de origem e da posição $m + 1$ até a posição $m + n$ referentes aos nós de destinos, a estratégia utilizada foi um pouco diferente. Para aproveitar, em rotinas de atualização da base, o fato de que sempre existe alternância entre origem e destino na árvore, foram dimensionadas na presente implementação duas variáveis do tipo **strNoArvore**, uma chamada ***Linha()*** (referente aos nós de origem) e outra chamada ***Coluna()*** (referentes aos nós de destino).

Para a árvore geradora apresentada na figura 43, que representa a solução básica factível da figura 40, o armazenamento das informações nas Variáveis ***Linha()*** e ***Coluna()***, vetores do tipo **strNoArvore**, poderia ser representado pelas estruturas das figuras 45 e 46.

Linha()		
posição	*Pai*	*Filhos*
1	0	{2}
2	2	{5}
3	2	{1 3 4}
4	2	{}
5	4	{}

FIGURA 45 – EXEMPLO DE VARIÁVEL DO TIPO **strNoArvore** PARA ARMAZENAMENTO DE INFORMAÇÕES REFRENTES AOS NÓS DE ORIGEM

FONTE: O Autor (2014)

Destaca-se que Linha(s).Pai=0 significa que o nó s é a raiz da árvore, ou seja, para o presente exemplo s=$\mathcal{O}_1$.

Coluna()		
posição	*Pai*	*Filhos*
1	3	{}
2	1	{4 2 3}
3	3	{}
4	3	{5}
5	2	{}

FIGURA 46 – EXEMPLO DE VARIÁVEL DO TIPO **strNoArvore** PARA ARMAZENAMENTO DE INFORMAÇÕES REFRENTES AOS NÓS DE DESTINO

FONTE: O Autor (2014)

Dada uma solução básica factível na forma da apresentada na variável ***ProblemaTransporte****.SolucaoInicial(,)*, as variáveis ***Linha()*** e ***Coluna()*** podem ser preenchidas por um processo de busca em largura ou profundidade.

4.1.4 Estrutura de dados para associação de variáveis

Para que, dada uma variável, básica não seja necessário realizar uma busca ao longo do vetor ***Basicas()*** para identificar em qual posição ela está, assim como para o caso de uma variável não básica, utilizou-se uma variável para completar o armazenamento de dados chamada $MatrizReferencia(,)$, tal que

$$MatrizReferencia(i,j) = x > 0 \Leftrightarrow \begin{cases} Basicas(x).Linha = i \\ Basicas(x).Coluna = j \end{cases}$$

$$MatrizReferencia(i,j) = x < 0 \Leftrightarrow \begin{cases} NaoBasicas(-x).Linha = i \\ NaoBasicas(-x).Coluna = j \end{cases}$$

A utilização da variável $MatrizReferencia(,)$ foi importante para facilitar o acesso a informações, evitando que fossem necessárias diversas buscas ao longo da execução do algoritmo.

Continuando com o mesmo exemplo das seções anteriores deste capítulo, a variável $MatrizReferencia(,)$ seria preenchida de acordo com os valores da figura 47.

$MatrizReferencia(,)$				
-1	1	-2	-3	-4
-5	6	-6	-7	4
2	7	3	8	-8
-9	5	-10	-11	-12
-13	-14	-15	9	-16

FIGURA 47 – EXEMPLO DE PREENCHIMENTO DA VARIÁVEL $MatrizReferencia(,)$
FONTE: O Autor (2014)

Ressalta-se que caso o problema de transporte fosse esparso, poderia ser atribuído o valor zero à $MatrizReferencia(i,j)$ para a inexistência da possibilidade de transporte de $\mathcal{O}_i$ para $\mathcal{D}_j$ e a variável ***NaoBasicas()*** teria dimensão $m.n-(m+n-1)-z$, onde z representa o número de ligações inexistentes no grafo bipartido associado ao problema.

4.2 PROCEDIMENTOS PARA A IMPLEMENTAÇÃO COMPUTACIONAL DO MÉTODO

Assim como para as estruturas de armazenamento de dados, os pseudocódigos apresentados nas seções a seguir são resultados de diversas tentativas e adaptações realizadas ao longo do desenvolvimento da presente tese.

Como os métodos de solução inicial são amplamente apresentados na literatura e apresentam tempo computacional pequeno quando comparado ao necessário para o método iterativo de melhoria de solução, omite-se aqui a explicação deles e considera-se que já é conhecida uma solução inicial básica factível para o problema.

4.2.1 Pseudocódigo geral para resolução do Problema de Transporte

O procedimento ResolveProblemaTransporte (quadro 12) possui como parâmetro de entrada uma variável ***ProblemaTransporte*** do tipo **strTransporte** que possui as informações de custos, ofertas e demandas do problema e também uma variável com uma solução inicial, a qual é uma matriz em que cada linha corresponde a uma variável básica e que nas colunas constam o valor da variável, a origem e o destino, respectivamente.

```
01  Procedimento ResolveProblemaTransporte(ProblemaTransporte)
02      AdaptaBasicas(ProblemaTransporte.SolucaoInicial)
03      AdaptaNaoBasicas(ProblemaTransporte.SolucaoInicial)
04      CriarArvore(ProblemaTransporte.SolucaoInicial)
05      CalculaCustosAtualizados()
06      (MaiorEconomia, RefMaiorEconomia)=EncontraMaiorEconomia()
07      Enquanto MaiorEconomia>0
08          PontoQuebra=EncontraQuebra(NaoBasicas(RefMaiorEconomia).Linha,
                                       NaoBasicas(RefMaiorEconomia).Coluna)
09          MudaArvore(RefMaiorEconomia, PontoQuebra)
10          (MaiorEconomia, RefMaiorEconomia)=EncontraMaiorEconomia(NaoBasicas)
11      Fim Enquanto
12  Fim Procedimento
```

QUADRO 12 – PROCEDIMENTO GERAL PARA RESOLUÇÃO DO PROBLEMA DE TRANSPORTE

FONTE: O Autor (2014)

Explicações dos procedimentos do algoritmo do quadro 12 serão detalhados a seguir. Os procedimentos das linhas 02 a 05 (do quadro 12) são responsáveis por realizar o preenchimento inicial das variáveis e estruturas, enquanto os seguintes pelo processo de melhoria da solução. Caso na função EncontraMaiorEconomia da linha 06 seja retornado que a variável *MaiorEconomia* seja maior do que 0, tem-se que a solução não é ótima e os códigos das linhas 07 a 11 realizam o processo iterativo de melhoria da solução.

O procedimento AdaptaBasicas, apresentado no quadro 13, recebe como parâmetro uma matriz contendo as informações referentes a uma solução básica factível e preenche a variável ***Básicas()*** e também parte da variável $MatrizReferencia(,)$.

```
01  Procedimento AdaptaBasicas(Solucao)
02      Para i=1 até NumOrigens+NumDestinos-1
03          Basicas(i).Linha=Solucao(i,2)
04          Basicas(i).Coluna=Solucao(i,3)
05          Basicas(i).Quantidade=Solucao(i,1)
06          Basicas(i).CustoVariavel=Transporte.MatrizCustoVariavel(Basicas(i).Linha,
                                                Basicas(i).Coluna)
07          MatrizReferencia(Solucao(i, 2), Solucao(i, 3)) = i
08      Próximo i
09  Fim Procedimento
```

QUADRO 13 – PROCEDIMENTO PARA PREENCHIMENTO INICIAL DA VARIÁVEL ***Basicas()*** E DE PARTE DA VARIÁVEL $MatrizReferencia(,)$

FONTE: O Autor (2014)

A linha 02 do código do quadro 12 executa o código do quadro 13 e faz com que a variável **Basicas()** assuma os valores apresentados na figura 41.

A variável *Solucao(,)*, parâmetro de entrada de AdaptaBasicas, contém uma solução inicial básica factível. Por isso, caso a variável $MatrizReferencia(,)$ na posição (i,j) esteja com o valor zero[22], a variável x_{ij} é não básica e deve estar presente na variável ***NaoBasicas()***. O procedimento AdaptaNaoBasicas (quadro 14) realiza o preenchimento da variável ***NaoBasicas()*** e também da variável $MatrizReferencia(,)$ nas posições apropriadas.

```
01  Procedimento AdaptaNaoBasicas()
02      aux=0
03      Para i=1 até NumOrigens
04          Para j=1 até NumDestinos
05              Se MatrizReferencia(i,j)=0 então
06                  aux=aux+1
07                  MatrizReferencia(i,j)=-aux
08                  NaoBasicas(aux).Linha=i
09                  NaoBasicas(aux).Coluna=j
10                  NaoBasicas(aux).CustoVariavel=Transporte.MatrizCustoVariavel(i, j)
11              Fim Se
12          Próximo j
13      Próximo i
14  Fim Procedimento
```

QUADRO 14 – PROCEDIMENTO PARA PREENCHIMENTO INICIAL DA VARIÁVEL ***NaoBásicas()*** E DE PARTE DA VARIÁVEL $MatrizReferencia(,)$

FONTE: O Autor (2014)

[22] Caso a presente implementação fosse adaptada para problemas esparsos poder-se-ia inicializar a variável *MatrizReferencia(,)* com um valor maior que $mn - (m + n - 1)$ e realizar a comparação com este valor na linha 05 do código.

A linha 03 do código do quadro 12 executa o código do quadro 14 e faz com que a variável ***NaoBasicas()*** assuma os valores apresentados na figura 42.

A criação da árvore é realizada na presente implementação por meio de uma busca em largura a partir do nó raiz, o qual foi sempre utilizado como sendo o $\mathcal{O}_1$. No procedimento CriarArvore as variáveis *LinhasDaColuna()*, *ColunasDaLinha()* são vetores de lista, ou seja, cada posição destas variáveis é uma variável do tipo Lista[23]. Ao final do procedimento as variáveis ***Linha()*** e ***Coluna()***, as quais são do tipo **strNoArvore**, armazenam todas as informações necessárias, para a presente implementação, referentes à árvore geradora que representa a solução básica factível.

```
01  Procedimento CriarArvore(Solucao)
02      Para p=1 até NumOrigens+NumDestinos-1
03          AdicionaNaLista(ColunasDaLinha(Solucao(p,2)), Solucao(p,3))
04          AdicionaNaLista(LinhasDaColuna(Solucao(p,3)), Solucao(p,2))
05      Próximo p
06      Atual=1
07      Linha(Atual).Pai=0
08      AdicionaNaFila(FilaDeLinhas, Atual)
09      Enquanto
10          Enquanto existir elemento em FilaDeLinhas
11              Atual=RetirarDaFila(FilaDeLinhas)
12              Para cada elemento h na lista ColunasDaLinha(Atual)
13                  AdicionaNaLista(Linha(Atual).Filhos, h)
14                  Coluna(h).Pai=Atual
15                  RemoveDaLista(LinhasDaColuna(h), Atual)
16                  AdicionaNaFila(FilaDeColunas, h)
17              Próximo h
18          Fim Enquanto
19          Enquanto existir elemento em FilaDeColunas
20              Atual= RetirarDaFila(FilaDeColunas)
21              Para cada elemento h na lista LinhasDaColuna(Atual)
22                  AdicionaNaLista(Coluna(Atual).Filhos, h)
23                  Linha(h).Pai=Atual
24                  RemoveDaLista(ColunasDaLinha(h), Atual)
25                  AdicionaNaFila(FilaDeLinhas, h)
26              Próximo h
27          Fim Enquanto
28      Fim Enquanto
29  Fim Procedimento
```

QUADRO 15 – PROCEDIMENTO PARA PREENCHIMENTO DE INFORMAÇÕES NAS VARIÁVEIS **Linha()** e **Coluna()** PARA REPRESENTAÇÃO DA ÁRVORE GERADORA ASSOCIADA A SOLUÇÃO BÁSICA FACTÍVEL

FONTE: O Autor (2014)

[23] Se ExLista é uma variável do tipo Lista e ExValor é um valor, AdicionaNaLista(ExLista, ExValor) adiciona ExValor em ExLista e RemoveDaLista(ExLista, ExValor) remove o ExValor de ExLista.

No capítulo 3 foi demonstrado que, conhecidos os custos atualizados da solução básica factível atual, não é necessário o cálculo das variáveis duais para a obtenção dos novos valores dos custos atualizados. Sobre os valores dos custos atualizados referentes à solução inicial não é necessário calcular os valores das variáveis duais, uma vez que poderia ser utilizado o método *Stepping Stone* para isso. Entretanto, foi computacionalmente mais rápido o cálculo dos custos atualizados iniciais pelo método u-v e, por isso, este procedimento foi executado na presente implementação para o cálculo dos valores iniciais de custos atualizados.

```
01  Procedimento CalculaCustosAtualizados()
02      Obter os valores das variáveis duais
03      Calcular os valores de custos atualizados para preencher
04              NaoBasicas().CustoAtualizado
05      Fim Procedimento
```

QUADRO 16 – PROCEDIMENTO PARA O CÁLCULO DAS VARIÁVEIS DUAIS COM BASE NAS VARIÁVEL ***Basicas()*** JÁ ESTANDO PREENCHIDA

FONTE: O Autor (2014)

A execução do procedimento CalculaCustosAtualizados gera alteração nos valores das variáveis ***Transporte****.VetorU()*,***Transporte****.VetorV()* e ***NaoBasicas()****.CustoAtualizado*. Para o presente exemplo, estas variáveis serão completadas com os valores já apresentados nas figuras 40 e 42.

A função EncontraMaiorEconomia, apresentada no quadro 17 e executada nas linhas 06 e 10 do código do quadro 12, é responsável por encontrar a variável não básica com o custo atualizado mais negativo e retornar qual o valor da maior economia unitária (o oposto do custo atualizado) e em qual posição do vetor ***NaoBasicas()*** ela se encontra.

```
01  Função (MaiorEconomia, Referencia)=EncontraMaiorEconomia()
02      MaiorEconomia=0
03      Para i=1 até NumOrigens+NumDestinos-1
04          Se NaoBasicas(i).CustoAtualizado<MaiorEconomia
05              MaiorEconomia= NaoBasicas(i).CustoAtualizado
06              ReferenciaMaiorEconomia=i
07          Fim Se
08      Próximo i
09      MaiorEconomia*= -1
10  Fim Função
```

QUADRO 17 - FUNÇÃO PARA IDENTIFICAR QUAL A VARIÁVEL NÃO BÁSICA QUE APRESENTA O CUSTO ATUALIZADO MAIS NEGATIVO

FONTE: O Autor (2014)

Na linha 06 do código do quadro 12, quando executada a função EncontraMaiorEconomia para o presente exemplo, o retorno será *MaiorEconomia*=1 e *RefMaiorEconomia*=16. Logo a variável $x_{5,5}$ é a que entrará na base.

Quando um arco associado a uma variável não básica é adicionado à arvore geradora forma-se um ciclo que precisa ser identificado para determinar a variável que deixa de ser básica. Para isso, o processo foi dividido em três partes:

1. Determinar os ancestrais dos vértices origem e destino (Procedimento CriaListaAncestrais apresentado no quadro 18)
2. Identificar o ancestral comum de maior profundidade e, consequentemente, determinar o ciclo (Procedimento EncontraPontoEncontro apresentado no quadro 19)
3. Das variáveis candidatas a saírem da base, determinar a que possui o menor valor (menor quantidade transportada associada) e escolhê-la como variável que deixará a base (linhas de código 04 a 19 do procedimento EncontraQuebra apresentado no quadro 20).

```
01  Procedimento CriaListaAncestrais(Origem, Destino)
02      AncestraisOrigem={}
03      AncestraisDestino={}
04      aux1=Origem
05      aux2=Destino
06      AdicionarNaLista(AncestraisOrigem, aux1)
07      Enquanto AncestraisOrigem(aux1).Pai ≠ 0
08          aux1=Linha(aux1).Pai
09          AdicionarNaLista(AncestraisOrigem, aux1)
10          aux1=Coluna(aux1).Pai
11          AdicionarNaLista(AncestraisOrigem, aux1)
12      Fim Enquanto
13      AdicionarNaLista(AncestraisDestino, aux2)
14      aux2=Coluna(aux2).Pai
15      AdicionarNaLista(AncestraisDestino, aux2)
16      Enquanto AncestraisOrigem(aux2).Pai ≠ 0
17          aux2=Linha(aux2).Pai
18          AdicionarNaLista(AncestraisDestino, aux2)
19          aux2=Coluna(aux2).Pai
20          AdicionarNaLista(AncestraisDestino, aux2)
21      Fim Enquanto
22      InverterOrdem(AncestraisOrigem)
23      InverterOrdem(AncestraisDestino)
24  Fim Procedimento
```

QUADRO 18 – PROCEDIMENTO RESPONSÁVEL POR CRIAR A LISTA DE ANCESTRAIS DE UM NÓ ORIGEM E DE UM NÓ DESTINO

FONTE: O Autor (2014)

O procedimento CriaListaAncestrais é responsável por encontrar a lista dos ancestrais da Origem e do Destino, sendo que elas são retornadas em ordem inversa. Para a implementação, o ciclo foi dividido em duas partes: a primeira do nó origem ao ancestral comum de maior profundidade e a segunda do nó destino ao ancestral comum de maior profundidade. Depois de executado o procedimento CriaListaAncestrais, tem-se

AncestraisOrigem={1 2 3 4 5} (55)

AncestraisDestino={1 2 2 5} (56)

O procedimento EncontraPontoEncontro é responsável por determinar qual é a posição (nas listas *AncestraisOrigem* e *AncestraisDestino*) do ancestral comum de maior profundidade (variável *PontoEncontro*). Uma vez obtidas estas informações o $\theta - loop$ torna-se conhecido, possibilitando identificar qual valor a variável que está entrando na base assumirá e qual deixará de ser básica.

```
01  Procedimento EncontraPontoEncontro()
02      PontoEncontro=0
03      Enquanto AncestraisOrigem(PontoEncontro)=AncestraisDestino(PontoEncontro)
04          PontoEncontro= PontoEncontro +1
05          Se PontoEncontro ≥Tamanho(AncestraisOrigem) ou
                    PontoEncontro ≥Tamanh(AncestraisDestino)
06              Sair Enquanto
07          Fim Se
08      Fim Enquanto
09  Fim Procedimento
```

QUADRO 19 – PROCEDIMENTO PARA ENCONTRAR POSICAO DO ANCESTRAL COMUM DE MAIOR PROFUNDIDADE

FONTE: O Autor (2014)

Depois de executado o procedimento EncontraPontoEncontro (quadro 19) para o presente exemplo, tem-se

PontoEncontro=2 (57)

Ressalta-se aqui que a variável *PontoEncontro* estar com valor 2 significa que o nó ancestral comum de maior profundidade está na segunda posição das listas

AncestraisOrigem e *AncestraisDestino*. Desta forma, para o caso dos valores nas expressões (55), (56) e (57), o $\theta - loop$ é $\{\mathcal{O}_5, \mathcal{D}_4, \mathcal{O}_3, \mathcal{D}_2, \mathcal{O}_2, \mathcal{D}_5, \}$.

Os procedimentos CriaListaAncestrais e EncontraPontoEncontro são utilizados nas linhas de código 02 e 03, respectivamente, na função EncontraQuebra apresentada no quadro 20.

```
01  Função EncontraQuebra(Origem, Destino)
02      CriaListaAncestrais(Origem, Destino)
03      EncontraPontoEncontro()
04      MinimoQtd=Infinito
05      Para i=Tamanho(AncestraisDestino)-1 até aux3 Passo -2
06          QtdAux=Basicas(MatrizReferencia(Elemento(AncestraisDestino, i - 1),
                    Elemento(AncestraisDestino, i))).Quantidade
07          SeQtdAux < MinimoQtd Então
08              MinimoQtd = QtdAux
09              Quebra=MatrizReferencia(Elemento(AncestraisDestino, i - 1),
                        Elemento(AncestraisDestino, i))
10          Fim Se
11      Próximo i
12      Para i=Tamanho(AncestraisOrigem)-1 até aux3 Passo -2
13          QtdAux=Basicas(MatrizReferencia(Elemento(AncestraisOrigem, i - 1),
                    Elemento(AncestraisOrigem, i))).Quantidade
14          SeQtdAux < MinimoQtd Então
15              MinimoQtd = QtdAux
16              Quebra=MatrizReferencia(Elemento(AncestraisOrigem, i - 1),
                        Elemento(AncestraisOrigem, i))
17          Fim Se
18      Próximo i
19      Retorna Quebra
20  Fim Função
```

QUADRO 20 – FUNÇÃO EncontraQuebra

FONTE: O Autor (2014)

Para exemplificar a função EncontraQuebra (apresentado no quadro 20) será utilizado como referência o mesmo exemplo. O resultado obtido após a execução das linhas 02 e 03 deste código já foram apresentadas nas expressões (55), (56) e (57).

O código das linhas 04 a 19 (do quadro 20) é então responsável por identificar qual a variável possível de deixar a base, no caminho entre o nó origem e o ancestral comum de maior profundidade, com o menor valor atual. Ou seja, o código das linhas 04 a 19 é responsável por identificar qual é e em qual posição no

vetor **Basicas()** está a variável que sairá da base. Para o presente exemplo, após executado a função EncontraPontoQuebra, tem-se

$$Quebra=7 \tag{58}$$

O retorno desta função é a posição no vetor **Basicas()** que se encontra a variável que deixa a base para a entrada da variável desejada. Com base na posição, é possível acessar diretamente as informações de quantidade e qual origem e destino ela é referente.

4.2.2 Atualização da árvore geradora

Dado um arco que sai da árvore e outro que entra, (3,2) e (3,5) respectivamente para o presente exemplo, faz-se necessária a atualização das variáveis ***Linha()*** e ***Coluna()*** que são responsáveis pela representação da árvore e das variáveis ***Basicas()***, ***NaoBasicas()*** e *MatrizReferencia(,)*

A explicação do procedimento MudaArvoreOtimizada, assim como aconteceu para o caso da função EncontraPontoQuebra, será aqui realizada em blocos, sendo apresentados os valores das variáveis após a execução de cada um deles.

Um procedimento alternativo ao MudaArvoreOtimizada é o MudaArvore (quadro 26), a diferença é que no primeiro são recalculados apenas os custos atualizados necessários conforme expressão (54) enquanto no segundo são calculadas todas as variáveis duais e recalculados todos os custos atualizados (método MODI).

No procedimento MudaArvoreOtimizada, após a atualização das quantidades no $\theta - loop$, uma primeira verificação deve ser realizada para identificar se o arco (r_s, t_s) que sai da árvore geradora pertence ao caminho dos ancestrais da origem ou do destino. Quando o referido arco pertence aos ancestrais da origem, tem-se que $Pai(r_s) = t_s$, já para quando pertence aos ancestrais do destino, tem-se que $Pai(t_s) = r_s$.

```
01  Procedimento MudaArvoreOtimizada(ReferenciaEntra, ReferenciaSai)
02      Atualizar Quantidades no θ − loop
03      Realizar atualização das informações de pai e filhos de cada vértice, conforme
            ideia apresentada na seção 2.2.5
04      Determinar a árvore ℋ
05      Atualizar as variáveis NaoBasicas e Basicas
06  Fim Procedimento
```

QUADRO 21 – PROCEDIMENTO MudaArvoreOtimizada

FONTE: O Autor (2014)

Como o procedimento AtualizaQuantidades é executado já no início do procedimento MudaArvoreOtimizada ele será apresentado antes da explicação detalhada do código para o procedimento de atualização da árvore (linha 03 quadro 21).

A entrada da variável (p, q) na base no lugar da variável (r_s, t_s) faz com que seja necessário atualizar os valores primais das variáveis básicas, ou seja, é necessário realizar adições e subtrações alternadas ao longo do ciclo formado pela adição do arco (p, q) à árvore. Para isso, é executado o procedimento AtualizaQuantidades (linha 02 do procedimento do quadro 21), apresentado no quadro 22.

```
01  Procedimento AtualizaQuantidades(Origem, Destino)
02      (AncestraisOrigem,AncenstraisDestinos)=EncontraListaAncestrais
03      Aux3=EncontraPontoEncontro(AncestraisOrigem,AncestraisDestino)
04      (Quebra, MinimoQtd)=EncontraPontoQuebra
05      Se mod(Tamanho(AncestraisDestino)-aux3)=0 então
06          Para i=Tamanho(AncestraisDestino) até aux3+1 passo -2
07              Basicas(MatrizReferencia(Elemento(AncestraisDestino, i - 1),
                    Elemento(AncestraisDestino, i))).Quantidade -= MinimoQtd
08              Basicas(MatrizReferencia(Elemento(AncestraisDestino, i - 1),
                    Elemento(AncestraisDestino, i - 2))).Quantidade += MinimoQtd
09          Próximo i
10          Para i=Tamanho(AncestraisOrigem) até aux3+1 passo -2
11              Basicas(MatrizReferencia(Elemento(AncestraisOrigem, i),
                    Elemento(AncestraisOrigem, i-1))).Quantidade -= MinimoQtd
12              Basicas(MatrizReferencia(Elemento(AncestraisOrigem, i - 2),
                    Elemento(AncestraisDestino, i - 1))).Quantidade += MinimoQtd
13          Próximo i
14          Basicas(MatrizReferencia(Elemento(AncestraisOrigem, aux3),
                    Elemento(AncestraisOrigem, aux3 - 1))).Quantidade -= MinimoQtd
15      Senão
16          Para i=Tamanho(AncestraisDestino) até aux3+1 passo -2
17              Basicas(MatrizReferencia(Elemento(AncestraisDestino, i - 1),
                    Elemento(AncestraisDestino, i))).Quantidade -= MinimoQtd
```

```
18              Basicas(MatrizReferencia(Elemento(AncestraisDestino, i - 1),
                    Elemento(AncestraisDestino, i - 2))).Quantidade += MinimoQtd
19          Próximo i
20          Basicas(MatrizReferencia(Elemento(AncestraisDestino, aux3-1),
                    Elemento(AncestraisDestino, aux3))).Quantidade -= MinimoQtd
21          Para i=Tamanho(AncestraisOrigem) até aux3+1 passo -2
22              Basicas(MatrizReferencia(Elemento(AncestraisOrigem, i),
                    Elemento(AncestraisOrigem, i-1))).Quantidade -= MinimoQtd
23              Basicas(MatrizReferencia(Elemento(AncestraisOrigem, i - 2),
                    Elemento(AncestraisDestino, i - 1))).Quantidade += MinimoQtd
24           Próximo i
25      Fim Se
26      Basicas(Quebra).Quantidade = MinimoQtd
27  Fim Procedimento
```

QUADRO 22 – PROCEDIMENTO AtualizaQuantidades

FONTE: O Autor (2014)

Retornando ao procedimento de atualização da árvore, tem-se para o exemplo que está sendo apresentado neste capítulo a situação descrita no quadro 23.

```
01  PaiOrigem=4
02  PaiDestino=5
03  Linha(5).Pai=5
04  Adiciona(Coluna(5).Filhos, 5)
05  Remove(Coluna(2).Filhos, 3)
06  OrigemAtual=5
07  Enquanto
08      5≠3? (Sim)
09          PaiOrigem2=2
10          PaiDestino2=1
11          Remove(Coluna(4).Filhos,5)
12          Coluna(4).Pai=5
13          Adiciona(Linha(5).Filhos, 4)
14          Adiciona(Coluna(4).Filhos, 3)
15          Linha(3).Pai=4
16          Remove(Linha(3).Filhos, 4)
17          OrigemAtual=3
18          PaiOrigem=2
19          PaiDestino=1
20      3≠3? Não
21  Fim Enquanto
```

QUADRO 23 – EXEMPLO PARA AS LINHAS 06 A 24 DO CÓDIGO DO PROCEDIMENTO MudaArvoreOtimizada

FONTE: O Autor (2014)

Para a explicação do código do quadro 23, considera-se que se a variável x_{pq} está entrando na base, dizer nó p significa dizer nó origem da variável que está entrando na base.

As atualizações das linhas 01 a 06 do código do quadro 23 ocorrem independentemente de quantos nós existem no caminho do nó p até o nó r_s. Já as atualizações das linhas 09 a 19 do mesmo quadro ocorrem para cada par de nós no caminho do nó p ao nó r_s. A atualização em pares permite que o teste lógico da linha seja realizado metade das vezes do que se fosse realizada a atualização individual.

Ainda, as atualizações das linhas 01 a 06 (quadro 23) são responsáveis por atualizar as informações dos nós t_s, q e pai do nó p. Já as linhas 09 a 19 (quadro 23) corrigem as informações dos nós no caminho de p a r_s, sendo repetidas $|\Omega(p, r_s)|/2$ vezes.

1. É armazenado na variável PaiOrigem o número do nó pai do nó p
2. É armazenado na variável PaiDestino o número do nó avô (nó pai do nó pai) do nó p.
3. O pai do nó p passa a ser o nó q
4. O nó p é adicionado a lista de filhos do nó q
5. O nó r_s é removido da lista de filhos do nó t_s
6. Na variável OrigemAtual é armazenado o valor do nó p
8. É realizado o teste lógico para verificar se ainda existem nós a serem atualizados
9. É armazenado na variável PaiOrigem2 o valor nó bisavô do nó OrigemAtual
10. É armazenado na variável PaiDestino2 o valor nó tataravô do nó OrigemAtual. PaiOrigem2 e PaiDestino armazenam as informações do próximo par de nós no caminho até a raiz.
11. O nó origem 5 deixa de ser filho do nó destino 4
12. O nó origem 5 passa a ser pai do nó destino 4
13. O nó destino 4 passa a ser filho do nó origem 5
14. O nó origem 3 passa a ser filho do nó destino 4
15. O nó destino 4 passa a ser pai do nó origem 3
16. O nó destino 4 deixa de ser filho do nó origem 3

17. OrigemAtual passa a ser o bisavô do nó OrigemAtual anterior para que o próximo par de nós seja atualizado
18. O valor de PaiOrigem é atualizado para ser o valor do nó pai do novo nó OrigemAtual
19. O valor de PaiDestino é atualizado para ser o valor do nó avô do novo nó OrigemAtual

Após a execução do procedimento do quadro 21, os novos valores das variáveis ***Linha()*** e ***Coluna()*** são os apresentados, respectivamente, nas figuras 48 e 49.

Linha()		
posição	*Pai*	*Filhos*
1	0	{2}
2	2	{5}
3	**4**	**{1 3}**
4	2	{}
5	**5**	**{4}**

FIGURA 48 – VARIÁVEL ***Linha()*** APÓS A MUDANÇA DE ÁRVORE CAUSADA PELA TROCA DE BASE

FONTE: O Autor (2014)

Coluna()		
posição	*Pai*	*Filhos*
1	3	{}
2	1	**{4** 2**}**
3	3	{}
4	**5**	**{3}**
5	2	**{5}**

FIGURA 49 – VARIÁVEL ***Coluna()*** APÓS A MUDANÇA DE ÁRVORE CAUSADA PELA TROCA DE BASE

FONTE: O Autor (2014)

Desta forma, a árvore geradora associada a nova solução básica factível passa a ser a apresentada na figura 50.

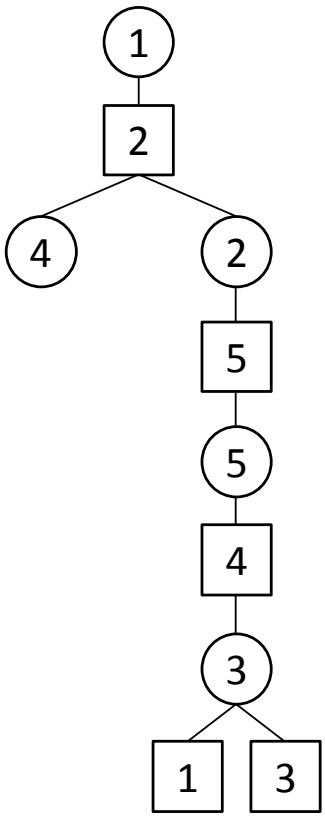


FIGURA 50 – ÁRVORE GERADORA ASSOCIADA À NOVA SOLUÇÃO
FONTE: O Autor (2014)

Além da atualização dos valores das variáveis ***Linha()*** e ***Coluna()*** que definem a árvore, também é necessária a atualização das variáveis ***Basicas()***, ***NaoBasicas()*** e *MatrizReferencia(,)*. Os valores destas variáveis atualizadas após a execução do procedimento MudaArvoreOtimizada são apresentados, respectivamente, nas figuras 51, 52 e 53. Destaca-se que estes são os valores após a execução do procedimento completo, já incluindo o procedimento DeterminaArvoreH que será apresentado após as referidas figuras e que os valores alterados em relação a iteração anterior estão destacados em negrito.

Basicas()					
Posição	*Linha*	*Coluna*	*Quantidade*	*CustoVariavel*	*CustoTotal*
1	1	2	22	3	66
2	3	1	21	3	63
3	3	3	6	3	18
4	2	5	**38**	4	**152**
5	4	2	7	4	28
6	2	2	**6**	5	**30**
7	**5**	**5**	4	**5**	**20**
8	3	4	**7**	6	**42**
9	5	4	**40**	7	**280**

FIGURA 51 – NOVOS VALORES DA VARIÁVEL ***Basicas()*** APÓS ATUALIZAÇÃO DAS QUANTIDADES GERADA PELA TROCA DE BASE

FONTE: O Autor (2014)

Destaca-se aqui que as informações de Custo Total não são atualizadas no procedimento implementado, uma vez que elas são desnecessárias. Mesmo assim, na figura 51 são apresentados os valores corretos caso esta informação fosse utilizada.

NaoBasicas()				
posição	*Linha*	*Coluna*	*CustoVariavel*	*CustoAtualizado*
1	1	1	7	**6**
2	1	3	4	**3**
3	1	4	4	**0**
4	1	5	7	5
5	2	1	5	**2**
6	2	3	5	**2**
7	2	4	7	**1**
8	3	5	5	**1**
9	4	1	3	**1**
10	4	3	7	**5**
11	4	4	7	**2**

12	4	5	5	2
13	5	1	7	3
14	5	2	7	**1**
15	5	3	4	0
16	**3**	**2**	**6**	**1**

FIGURA 52 – NOVOS VALORES DA VARIÁVEL ***NaoBasicas()*** APÓS ATUALIZAÇÃO POR CAUSA DA TROCA DE BASE

FONTE: O Autor (2014)

Para exemplo pequenos, como o do presente capítulo, pode não parecer de grande vantagem o recálculo dos custos atualizados pela expressão (54), uma vez que 12 das 16 posições na variável ***NaoBasicas()*** tiveram valores de *CustoAtualizado* alterados. Entretanto, para problemas maiores, o percentual de custos atualizados não recalculados é na média maior e, além disso, pela expressão (54) para os custos atualizados que necessitam ser recalculados é realizada apenas uma operação, enquanto para o cálculo pelo método MODI são necessárias duas operações.

$MatrizReferencia(,)$				
-1	1	-2	-3	-4
-5	6	-6	-7	4
2	**-16**	3	8	-8
-9	5	-10	-11	-12
-13	-14	-15	9	**7**

FIGURA 53 – VARIÁVEL $MatrizReferencia(,)$ APÓS O FIM DA ITERAÇÃO

FONTE: O Autor (2014)

O procedimento de determinar a árvore $\mathcal{H}$ (utilizado no procedimento MudaArvoreOtimizada do quadro 21) é apresentado no quadro 24 e após ele ser executado, os valores armazenados nas variáveis *OrigemEmH*, *DestinoEmH*, *OrigemNaoEmH* e *DestinoNaoEmH* tornam possível identificar quais são as variáveis não básicas que possuem alteração no valor de custo atualizado. O cálculo dos novos valores é então realizado pelo procedimento AtualizaCustosAtualizados, apresentado no quadro 25.

```
01  Procedimento DeterminaArvoreH()
02      Enquanto existir elemento em ListaOrigem ou ListaDestino
03          Se existe elemento em ListaOrigem então
04              Para cada origem em ListaOrigem
05                  Para cada destino em Linha(origem).Filhos
06                      Adiciona(ListaDestino, destino)
07                      Adiciona(DestinoEmH, destino)
08                      Remove(DestinoNaoEmH, destino)
09                  Próximo destino
10              Próxima origem
11              Limpa(ListaOrigem)
12          Fim Se
13          Se existe elemento em ListaDestino então
14              Para cada destino em ListaDestino
15                  Para cada origem em Coluna(destino).Filhos
16                      Adiciona(ListaOrigem, origem)
17                      Adiciona(OrigemEmH, origem)
18                      Remove(OrigemNaoEmH, origem)
19                  Próximo destino
20              Próxima origem
21              Limpa(ListaDestino)
22          Fim Se
23      Fim Enquanto
24  Fim Procedimento
```

QUADRO 24 – PROCEDIMENTO DeterminaArvoreH

FONTE: O Autor (2014)

Para o presente exemplo, tem-se, após a execução do procedimento DeterminaArvoreH que

$$\text{OrigemEmH}=\{3\ 5\} \tag{59}$$

$$\text{OrigemNaoEmH}=\{1\ 2\ 2\} \tag{60}$$

$$\text{DestinoEmH}=\{1\ 3\ 4\} \tag{61}$$

$$\text{DestinoNaoEmH}=\{2\ 5\} \tag{62}$$

Uma vez identificados os vértices que pertencem às árvores $\mathcal{H}$ e $\mathcal{T}$-$\mathcal{H}$ pelos valores das variáveis *OrigemEmH*, *OrigemNaoEmH*, *DestinoEmH* e *DestinoNaoEmH*, é executado o procedimento AtualizaCustosAtualizados.

```
01  Procedimento AtualizaCustosAtualizados()
02      Para cada i em OrigensEmH
03          Para cada j em DestinosNaoEmH
04              NaoBasicas(-MatrizReferencia(i, j)).CustoAtualizado += Alfa * Cpq
05          Próximo j
06      Próximo i
07      Para cada i em OrigensNaoEmH
08          Para cada j em DestinosEmH
09              NaoBasicas(-MatrizReferencia(i, j)).CustoAtualizado -= Alfa * Cpq
10          Próximo j
11      Próximo i
12  Fim Procedimento
```

QUADRO 25 – PROCEDIMENTO AtualizaCustosAtualizados

FONTE: O Autor (2014)

O procedimento AtualizaCustosAtualizados calcula o novo valor de cada um dos custos atualizados necessário, de acordo com o demonstrado no capítulo 3. Neste caso, sendo $m_{\mathcal{H}}$ o número de origens em $\mathcal{H}$ e $n_{\mathcal{H}}$ o número de destinos em $\mathcal{H}$, o número de recálculos necessários é de $\left(m_{\mathcal{H}}(n - n_{\mathcal{H}})\right) + \left(n_{\mathcal{H}}(m - m_{\mathcal{H}})\right)$.

O procedimento AtualizaCustosAtualizados faz com que não seja necessário o recálculo de todos os custos atualizados. Assim, quando grande parte dos nós (origem e destino) pertencem simultaneamente à árvore $\mathcal{H}$ (ou não pertencem simultaneamente) o percentual de custos atualizados que precisam ser recalculados torna-se pequeno.

Para o presente exemplo, os valores da variável **NaoBasicas()** apresentado na figura 52 já contemplam os novos valores de custos atualizados. No exemplo tem-se que $m_{\mathcal{H}} = 2$ e $n_{\mathcal{H}} = 3$ e o número de custos atualizados recalculados é $2x(5 - 2) + 3x(5 - 3) = 12$, ou seja, 12 custos atualizados são recalculados.

O procedimento MudaArvore (quadro 26) possui características semelhantes ao procedimento MudaArvoreOtimizada, sendo que a diferença, conforme já explicado, é no cálculo dos novos valores de custos atualizados. Como todos os custos atualizados são recalculados em MudaArvore, não faz-se necessária a determinação da árvore $\mathcal{H}$. Além disso, o procedimento AtualizaCustosAtualizados é substituído pelo procedimento CalculaCustosAtualizados.

01	Procedimento MudaArvore (ReferenciaEntra, ReferenciaSai)
02	Atualizar Quantidades no $\theta - loop$
03	Realizar atualização das informações de pai e filhos de cada vértice, conforme ideia apresentada na seção 2.2.5
04	Calcula os novos valores das variáveis NaoBasicas (incluindo o recálculo de
05	todos os custos atualizados conforme método MODI) e Basicas
06	Fim Procedimento

QUADRO 26 – PROCEDIMENTO MudaArvore

FONTE: O Autor (2014)

Com a implementação realizada em árvore utilizando o procedimento MudaArvoreOtimizada, a parte que tornou-se a mais demorada foi o procedimento EncontraMaiorEconomia, pois nele é realizada a busca por todas as variáveis não básicas para encontrar a que possui o custo atualizado mais negativo, sendo necessárias $m.n-(m+n-1)$, ou n^2-2n+1 para problemas quadrados, comparações.

Este era o único procedimento que o número de operações aproximava-se de n^2 e, portanto, passou a ser estudado como a situação poderia ser melhorada. A primeira ideia foi então a utilização do *partial pricing*, ou seja, a escolha da primeira variável não básica com custo atualizado negativo encontrada como sendo a variável a entrar na base. Entretanto, este conceito levou a um tempo de resolução maior, uma vez que começaram a ser escolhidas variáveis para entrar na base que geravam pouca redução no valor da função objetivo e o tempo gasto pelo maior número de iterações necessárias não era compensado pelo ganho de tempo na decisão da variável a entrar na base. Além disso, foram testadas divisões em grupos e os tempos de resolução também foram piores.

4.3 NOVO CRITÉRIO PARA ESCOLHA DA VARIÁVEL A TORNAR-SE BÁSICA

Diante do tempo gasto na escolha da variável a entrar na base e do mau resultado da utilização do *Partial Pricing*, foi criado um procedimento de armazenar uma lista de variáveis com custos atualizados negativos, no qual após o primeiro cálculo dos valores de custos atualizados as posições no vetor ***NaoBasicas()*** das variáveis com custos atualizados negativos eram armazenadas na lista

ReferenciaCAN. Para exemplo da criação da lista *ReferenciaCAN* serão utilizados como base os valores da variável ***NaoBasicas()*** apresentados na figura 54

NaoBasicas()															
Posicao	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CustoAtualizado	3	-2	8	1	-7	2	5	-5	-1	-4	7	3	-1	6	2

FIGURA 54 – EXEMPLO DA VARIÁVEL ***NaoBasicas()*** PARA A CRIAÇÃO DA LISTA *ReferenciaCAN*
FONTE: O Autor (2014)

Para a variável não básicas apresentada na figura 54, a lista *ReferenciaCAN* obtida seria $\{2, 5, 8, 9, 10, 13\}$.

Nas iterações seguintes o processo de busca da variável não básica com o custo atualizado mais negativo seria então realizado somente nas variáveis relacionadas aos elementos de *ReferenciaCAN*. Quando não existissem mais elementos em *ReferenciaCAN* que estivessem associados à variáveis não básicas com custos atualizados negativos, a lista *ReferenciaCAN* seria limpa e realizava-se novamente uma busca das variáveis não básicas que possuíam custo atualizado negativo e então a posição na variável ***NaoBasicas()*** de cada variável não básica com custo atualizado negativo era armazenada na lista *ReferenciaCAN*, com processo idêntico ao do exemplo apresentado anteriormente para criação da lista *ReferenciaCAN*. Uma vez obtida a nova lista *ReferenciaCAN*, a busca pela variável não básica com o custo atualizado mais negativo ocorreria somente nas variáveis referenciadas pelos elementos desta lista.

A utilização da lista *ReferenciaCAN* conforme metodologia aqui exposta gerou uma nova redução de tempo computacional tanto para a implementação em árvore recalculando todos os custos atualizados como para a implementação em árvore recalculando somente os custos atualizados necessários.

Porém, mesmo que a utilização da lista *ReferenciaCAN* priorizasse a entrada de variáveis com custos atualizados mais negativos dentre as listadas, ocorriam casos de na lista *ReferenciaCAN* existirem somente referências à variáveis não básicas com custos atualizados negativos maiores do que o de outras variáveis não listadas com custos atualizados bem mais negativos. Um exemplo de evolução

dos custos atualizados negativos utilizados a cada iteração na resolução de um problema 400x400 é apresentado no gráfico 1.

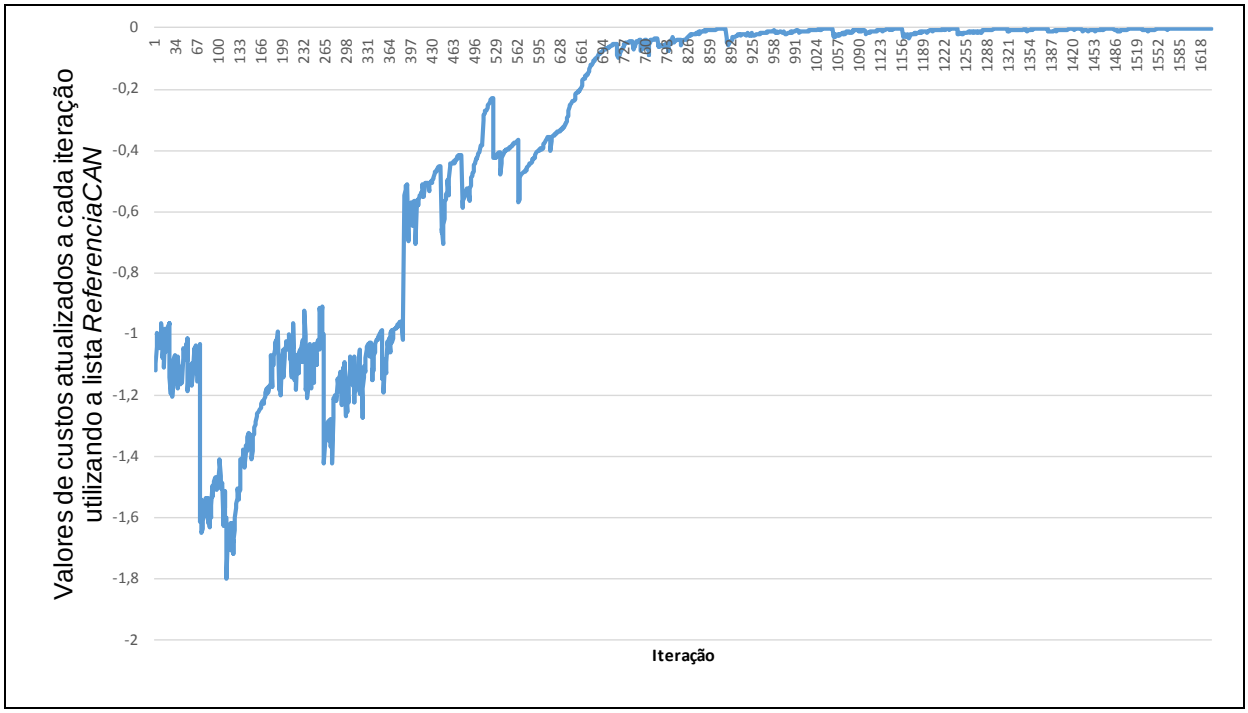


GRÁFICO 1 – EXEMPLO DE EVOLUÇÃO DOS VALORES DE CUSTOS ATUALIZADOS UTILIZADOS NUM EXEMPLO 400X400 COM A LISTA *ReferenciaCAN*

FONTE: O Autor (2014)

Diante disso, estudou-se a implantação de um novo parâmetro, denominado *PercentualEconomiaAnterior*, no método. Este parâmetro teve como objetivo fazer com que quando ocorresse um decréscimo acentuado na razão entre o custo atualizado da variável que entrou na base na iteração anterior e o custo atualizado mais negativo disponível nas variáveis referenciadas em *ReferenciaCAN*, a lista *ReferenciaCAN* fosse reiniciada.

Como exemplo da utilização do parâmetro *PercentualEconomiaAnterior*, será considerado o caso que a solução inicial possui os valores apresentados na figura 54 para a variável ***NaoBasicas()*** e, consequentemente, *ReferenciaCAN* sendo $\{2, 5, 8, 9, 10, 13\}$.

Neste caso a variável que entra na base é a que está na posição 5 da variável ***NaoBasicas()***. Desta forma, sejam os valores apresentados na figura 55 os novos valores da variável ***NaoBasicas()*** após a troca de base, a lista *ReferenciaCAN* é $\{2, 8, 9, 10, 13\}$ e a nova maior economia listada é -5. Se -5 for

menor do que *PercentualEconomiaAnterior* x (-7), então a lista *ReferenciaCAN* é reiniciada e passa a ser $\{7, 8, 9, 10, 13, 14\}$, caso contrário a variável da posição 8 do vetor ***NaoBasicas()*** entra na base e o processo continua iterativamente.

NaoBasicas()															
Posicao	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CustoAtualizado	3	5	8	1	7	2	-2	-5	-1	-4	7	10	-1	-1	2

FIGURA 55 – EXEMPLO DA VARIÁVEL ***NaoBasicas()*** PARA APÓS A TROCA DE BASE
FONTE: O Autor (2014)

Ressalta-se que é possível e comum, assim como aconteceu para o presente exemplo, de que variáveis não básicas com custos atualizados negativos ficassem fora da lista *ReferenciaCAN* e variáveis não básicas com custos atualizados positivos pertencessem à lista *ReferenciaCAN*. Entretanto, realizar o processo de retirar de *ReferenciaCAN* as variáveis não básicas com custos atualizados positivos, além de exigir tempo, pode fazer com que a lista *ReferenciaCAN* torne-se muito pequena e precise ser reinicializada mais vezes. Ao serem mantidas as variáveis com custos atualizados positivos pode acontecer das mesmas voltarem a assumir valores de custos atualizados negativos nas iterações seguintes.

O procedimento ResolveProblemaTransporteListaCAN, apresentado no quadro 27, é uma adaptação do procedimento ResolveProblemaTransporte (quadro 12) para a nova metodologia de escolha da variável a entrar na base a cada iteração.

```
01 Procedimento ResolveProblemaTransporteListaCAN(ProblemaTransporte)
02     AdaptaBasicas(ProblemaTransporte.SolucaoInicial)
03     AdaptaNaoBasicas(ProblemaTransporte.SolucaoInicial)
04     CriarArvore(ProblemaTransporte.SolucaoInicial)
05     CalculaCustosAtualizados()
06     ReferenciaCAN=CriaListaReferenciaCAN()
07     (MaiorEconomia, RefMaiorEconomia)=EMEconomiaReferenciaCAN()
08     Enquanto MaiorEconomia>0
09         MaiorEconomiaAnterior= -NaoBasicas(RefMaiorEconomia).CustoAtualizado
10         PontoQuebra=EncontraQuebra(NaoBasicas(RefMaiorEconomia).Linha,
11                                    NaoBasicas(RefMaiorEconomia).Coluna)
12         MudaArvore(RefMaiorEconomia, PontoQuebra)
13         Remove(ReferenciaCAN, RefMaiorEconomia)
14         (MaiorEconomia, RefMaiorEconomia)=EMEconomiaReferenciaCAN()
15         Se MaiorEconomia<PercentualEconomiaAnterior*EconomiaAnterior ou
16                                    MaiorEconomia=0 então
17             ReferenciaCAN=CriaListaReferenciaCAN()
18             (MaiorEconomia, RefMaiorEconomia)=EMEconomiaReferenciaCAN()
19             MaiorEconomiaAnterior=0
20          Fim Se
21     Fim Enquanto
22 Fim Procedimento
```

QUADRO 27 – PROCEDIMENTO ResolveProblemaTransporteListaCAN

FONTE: O Autor (2014)

A análise para os resultados gerados para diferentes valores de *PercentualEconomiaAnterior* é apresentada na seção 5.5. Destaca-se que definir *PercentualEconomiaAnterior* =0 significa fazer com que a lista seja reinicializada somente quando não existirem mais variáveis com custos atualizados negativos e os resultados para esta situação são apresentados na seção 5.4.

O objetivo do presente capítulo foi apresentar de forma detalhada a implementação realizada para que ela possa ser reproduzida por outros interessados. Ressalta-se que tanto a estrutura de dados utilizada, como o resultado matemático de recálculo somente dos custos atualizados necessários, as inclusões da lista *ReferenciaCAN* e do parâmetro *PercentualEconomiaAnterior* foram importantes, conforme resultados experimentais realizados, para a redução do tempo de resolução do Problema de Transporte.

No capítulo seguinte são ainda apresentados resultados para a implementação aqui descrita quando utilizada para a resolução de problemas de designação. Destaca-se que problemas de designação são de grande aplicação em diversos problemas e muitas vezes o interesse é resolver problemas retangulares.

Uma situação na qual problemas de designação retangulares[24] necessitam ser resolvidos diversas vezes é na alocação de equipes de atendimento. Por exemplo numa rede de energia elétrica onde existem m equipes para atender n ocorrências, onde $n > m$, onde existe um custo associado ao atendimento de cada ocorrência por cada equipe, como deslocamento, qualificação técnica, etc., problemas de designação são subproblemas que necessitam ser resolvidos diversas vezes.

Problemas de designação retangulares também existem no caso de alocação de pessoas para tarefas, como por exemplo a definição de tripulação para diferentes voos numa semana poderia ser resolvido por uma heurística que gerasse diversos problemas de designação. Neste caso a semana poderia ser dividida em subintervalos e os custos de alocação no subintervalo $i + p$ estão associados as alocações realizadas nos subintervalos $i + p - 1, i + p - 2, 1$.

Além disso, diversas outras aplicações nas quais os problemas de designação são subproblemas podem vir a ser realizadas e a viabilidade do tempo de solução dependerá dos algoritmos existentes. Por isso, embora não seja um algoritmo específico para designação, como o algoritmo húngaro, são analisados os tempos computacionais da implementação aqui apresentada para problemas de designação.

[24] Uma adaptação do algoritmo de designação para problemas retangulares é apresentada por Bourgeois e Lasalle (1971).

5 RESULTADOS COMPUTACIONAIS

No capítulo 4 foi detalhada a implementação para a estrutura em árvore e também apresentada a proposta da utilização da lista *ReferenciaCAN* referente às variáveis não básicas com custos atualizados negativos e do parâmetro *PercentualMaiorEconomia*.

No presente capítulo são apresentados e comparados os resultados computacionais para problemas gerados com valores aleatórios e resolvidos pelos métodos implementados. Para todos os exemplos foi utilizada como solução básica factível inicial a solução obtida pelo método do custo mínimo.

As implementações foram realizadas na linguagem VB.NET com projetos do tipo Windows Form Application. Os tempos apresentados neste capítulo são do código sendo executado no próprio ambiente de desenvolvimento.

5.1 COMPARAÇÃO ENTRE UTILIZAÇÃO DE ESTRUTURAS

A primeira comparação de tempo de resolução foi referente à estrutura de armazenamento de informações. Para isso foi comparada a clássica implementação em estrutura de quadro, com o $\theta - loop$ sendo identificado pelo procedimento apresentado por Souza (2004), com a implementação em estrutura de árvore utilizando o procedimento MudaArvore apresentado no quadro 26.

Nas duas implementações analisadas nesta seção o recálculo dos custos atualizados foi realizado pelo cálculo dos valores das variáveis duais, $u_i's$ e $v_j's$, e cálculo dos valores de custos atualizados de todas as variáveis não básicas por $\bar{c}_{ij} = c_{ij} - u_i - v_j$, ou seja, são duas implementações do método MODI utilizando duas diferentes estruturas de armazenamento de dados que possibilitam diferentes procedimentos, uma em quadro e outra em árvore.

Como foi observado por Silva (2012) que, para um mesmo número de variáveis, os problemas quadrados são os que necessitam de mais iterações e

possuem maiores tempos de resolução, foram gerados para testes problemas de dimensão $100x100$, $200x200$, $300x300$ e $400x400$.

5.1.1 Comparação entre utilização de estruturas para Problemas de Transporte

Os valores de ofertas e demandas foram valores inteiros entre 300 e 800 por também terem sido os valores utilizados por Silva (2012). Para analisar também se a precisão dos custos teria influência no número de iterações e tempo de resolução foram considerados também exemplos com diferentes intervalos de custos. Quando os custos estão no intervalo de 1 a 10, estão sendo considerados custos com um dígito de precisão, quando estão no intervalo de 1 a 100, estão sendo considerados dois dígitos de precisão e assim por diante.

Quando os custos são valores não subjetivos é difícil (e até impossível) ter controle sobre os valores deles, entretanto para o caso de criação de matriz de custos por processos subjetivos (ou que não necessitem de grande precisão) pode ser considerado um limite no número de custos possíveis para o preenchimento da matriz e, por isso, é aqui realizada a análise para diferentes possibilidades de precisão dos valores dos custos.

Na tabela 1 é apresentado o tempo médio de resolução, em milissegundos, a economia de tempo da resolução em estrutura de árvore em relação a estrutura em quadro e o número médio de iterações para 100 problemas de cada configuração de parâmetros. Ressalta-se que os mesmos problemas, com as mesmas soluções iniciais, foram utilizados para as duas diferentes estruturas.

No presente capítulo, os resultados para a implementação do método MODI em quadro serão referenciados como Q_MODI e para o caso da implementação do método MODI em árvore como A_MODI.

TABELA 1 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO PARA AS ESTRUTURAS EM QUADRO E EM ÁRVORE

Dimensão	Custo Máximo	Tempo médio de resolução A_MODI (ms)	Tempo médio de resolução Q_MODI (ms)	Economia de A_MODI em relação à Q_MODI	Número médio de iterações
100	10	48,29	161,34	70,07%	88,32
100	100	112,55	278,57	59,60%	204,61
100	1000	114,41	282,43	59,49%	207,54
100	10000	113,61	281,50	59,64%	206,08
200	10	248,26	882,43	71,87%	116,93
200	100	1.136,43	2.620,14	56,63%	533,75
200	1000	1.127,02	2659,75	57,63%	529,27
200	10000	1.129,31	2.666,70	57,65%	530,34
300	10	640,77	2.095,86	69,43%	136,05
300	100	4.314,78	9.849,63	56,19%	908,32
300	1000	4.338,37	9.869,01	56,04%	912,91
300	10000	4.292,83	9.768,43	56,05%	903,25
400	10	1.311,58	4.020,09	67,37%	156,77
400	100	9.917,43	23.490,95	57,78%	1.171,88
400	1000	11.220,34	24.532,77	54,26%	1.323,63
400	10000	11.288,12	24.620,87	54,15%	1.331,85

FONTE: O Autor (2014)

Com base nos valores da tabela 1 observa-se que o número de dígitos de precisão tem influência no tempo de resolução. Embora em diversos problemas não seja possível limitar a precisão da matriz de custos, esta análise pode ser interessante para a modelagem de novos problemas, principalmente onde os custos são definidos de forma subjetiva e a limitação da precisão deles para um dígito seja viável. A partir da seção 5.3 são apresentados testes com variáveis do tipo ponto flutuante de simples precisão.

A razão economia de tempo entre o tempo médio necessário para a obtenção da solução ótima pela estrutura em árvore e o tempo médio necessário para a obtenção da solução ótima pela estrutura em quadro foi entre 54,15% e 71,87% (tabela 1). Considerando a média para todos os exemplos testados, a economia foi de 60,24%.

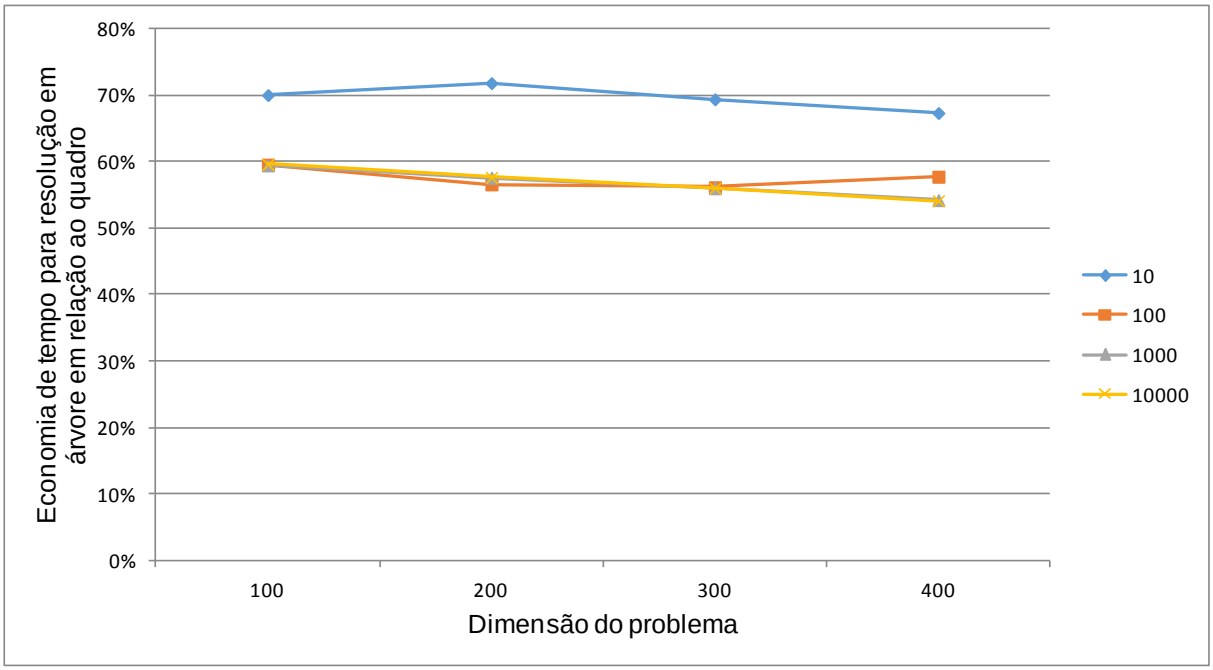


GRÁFICO 2 – COMPARAÇÃO DO PERCENTUAL DE ECONOMIA DE TEMPO DA RESOLUÇÃO EM ÁRVORE EM RELAÇÃO À RESOLUÇÃO EM QUADRO PARA O PROBLEMA DE TRANSPORTE

FONTE: O Autor (2014)

As maiores economias de tempos geradas pela utilização da estrutura em árvore em relação à estrutura em quadro ocorreram para a menor das amplitudes de custos testadas (gráfico 2), ou seja, para o caso de custos com 1 digito de precisão. Sobre o efeito da dimensão do problema no percentual de economia, não foi possível observar padrão de comportamento (crescente ou decrescente) em função do aumento das dimensões dos problemas.

5.1.2 Comparação entre utilização de estruturas para Problemas de Designação

A mesma ideia de análise da seção 5.1.1 foi realizada para problemas nos quais as ofertas e demandas são iguais a 1, ou seja, problemas de designação. Os resultados de tempos computacionais obtidos, em milissegundos, são apresentados na tabela 2.

TABELA 2 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO PARA AS ESTRUTURAS EM QUADRO E EM ÁRVORE

Dimensão	Custo Máximo	Tempo médio de resolução A_MODI (ms)	Tempo médio de resolução Q_MODI (ms)	Economia de A_MODI em relação à Q_MODI	Número médio de iterações
100	10	78,03	387,95	79,89%	149,37
100	100	179,16	590,39	69,65%	340,04
100	1000	179,36	607,35	70,47%	340,56
100	10000	180,96	613,20	70,49%	343,05
200	10	486,66	3.127,18	84,44%	244,50
200	100	1.792,93	5.988,04	70,06%	880,46
200	1000	1.793,87	6.196,15	71,05%	880,81
200	10000	1.800,37	6.168,05	70,81%	884,40
300	10	1.532,43	10.846,49	85,87%	348,66
300	100	6.752,16	22.100,04	69,45%	1.484,90
300	1000	6.880,95	22.441,72	69,34%	1.511,97
300	10000	6.908,53	22.356,50	69,10%	1.517,70
400	10	3.574,35	25.698,91	86,09%	460,86
400	100	15.317,82	52.342,36	70,74%	1.897,79
400	1000	18.135,97	56.923,05	68,14%	2.239,24
400	10000	18.242,47	57.251,85	68,14%	2.249,93

FONTE: O Autor (2014)

Para os problemas de designação foi então observado um ganho de 68,14% a 86,09% no tempo de resolução (tabela 2) quando utilizada a estrutura em árvore em relação à estrutura em quadro na implementação do método MODI. O ganho médio considerando todos os exemplos testados foi de 73,36%.

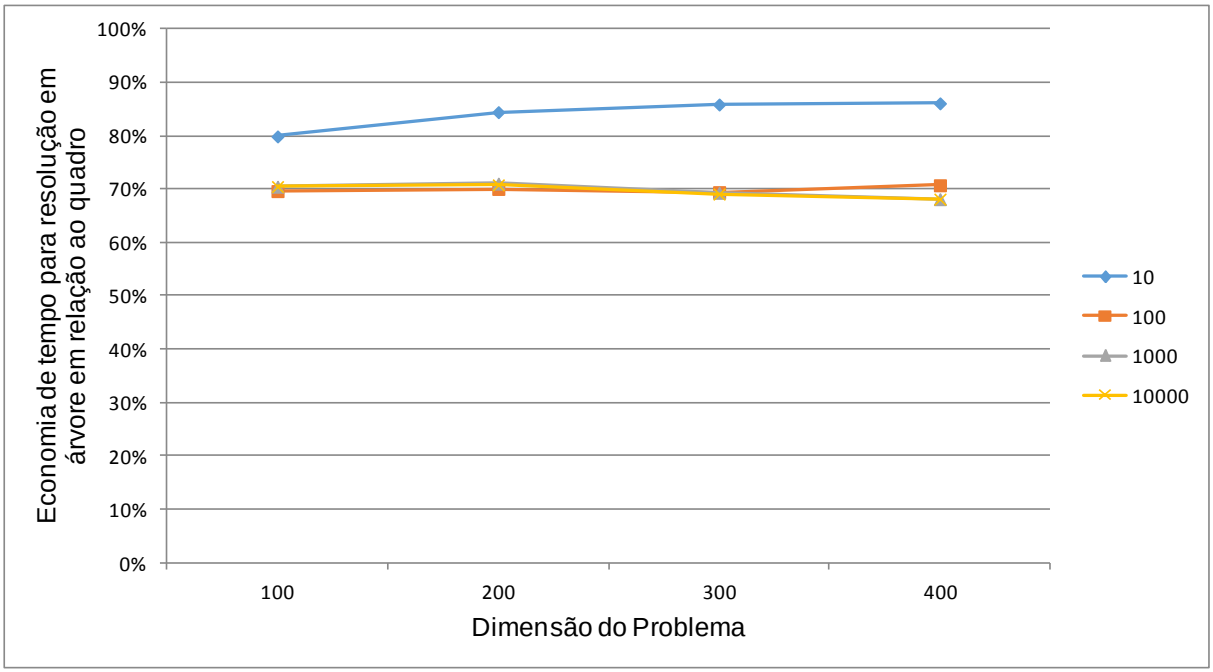


GRÁFICO 3 – COMPARAÇÃO DO PERCENTUAL DE ECONOMIA DE TEMPO DA RESOLUÇÃO EM ÁRVORE EM RELAÇÃO À RESOLUÇÃO EM QUADRO PARA PROBLEMAS DE DESIGNAÇÃO

FONTE: O Autor (2014)

Assim como para problemas com ofertas e demandas entre 300 e 800, as maiores economias de tempos para problemas de designação geradas pela utilização da estrutura em árvore em relação à estrutura em quadro ocorreram para a situação da menor número de dígitos de precisão para os custos testada (gráfico 3).

Além disso, não foi possível, para as dimensões testadas, identificar algum padrão de crescimento ou decrescimento do percentual de economia em função do número de origens e destinos de problemas quadrados. Entretanto, não é possível garantir que o mesmo comportamento também ocorre para outras dimensões de problemas.

Com base nos resultados analisados nas tabelas 1 e 2, constata-se a importância que a estrutura de dados possui na implementação do mesmo método para a resolução do Problema de Transporte e a vantagem, em termos de tempo de resolução, da utilização da estrutura em árvore.

Embora fosse esperado que a estrutura de árvore proporcionasse um tempo de resolução menor devido ao menor número de operações para encontrar o $\theta - loop$, esta análise foi importante para medir o tamanho da economia e assim

exemplificar a importância da estrutura de dados apropriada para a resolução de um Problema de Transporte.

5.1.3 Operações necessárias para a identificação do $\theta - loop$ na estrutura de árvore

Para a identificação do $\theta - loop$ na estrutura em árvore utiliza-se a ideia de encontrar as listas dos ancestrais dos nós de origem e destino e, uma vez encontradas as duas listas, identificar o ancestral comum de maior profundidade. No pior caso, para $m = n$, possível (figura 56) a soma dos tamanhos das duas listas será $2(m + n) - 3$ e o ancestral comum de maior profundidade é encontrado depois de $m + n - 2$ verificações.

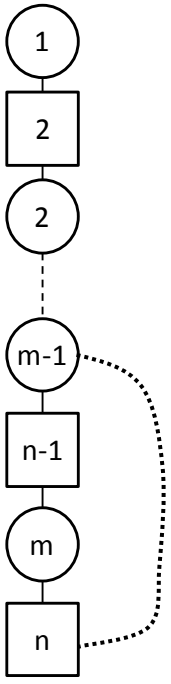


FIGURA 56 – SITUAÇÃO DA ÁRVORE GERADORA PARA O PIOR CASO PARA IDENTIFICAÇÃO DO $\theta - loop$

FONTE: O Autor (2014)

Na figura 56 é ilustrada a situação do pior caso para a identificação do $\theta - loop$, sendo que para a entrada da variável x_{m-1}, n na base, a lista de ancestrais do nó $\mathcal{O}_{m-1}$ possui $m + n - 3$ elementos e a lista dos ancestrais de $\mathcal{D}_n$ possui $m + n$

elementos, totalizando $2(m+n)-3$ elementos nas duas listas. Ainda neste caso, o ancestral comum de maior profundidade é o próprio nó $\mathcal{O}_{m-1}$, sendo necessárias $m+n-2$ verificações.

Porém, este é o comportamento no pior caso e não necessariamente é o que ocorre. Diante disso, para os exemplos do com matriz de custos, ofertas e demandas do tipo ponto flutuante de simples precisão, apresentados a partir da seção 5.3, foram coletadas informações sobre o tamanho das listas de ancestrais e de quantas comparações foram necessárias para encontrar o ancestral comum de maior profundidade.

Uma análise mais detalhada sobre o $\theta - loop$ é realizada na seção 5.3.3 após serem apresentados os resultados para variáveis do tipo ponto flutuante de simples precisão.

Para os casos de problemas de designação com variáveis do tipo ponto flutuante de simples precisão de tamanhos $100x100$, $200x200$, $300x300$ e $400x400$, a média da soma do número de elementos (tamanho) das duas listas de ancestrais foram, respectivamente, de 37,78, 55,43, 75,70 e 90,61, enquanto que os números médios de comparações para a identificação do ancestral comum de maior profundidade foram de 5,59, 6,70, 8,43 e 8,37.

Já para os casos de problemas de designação com variáveis do tipo ponto flutuante de simples precisão de tamanhos $100x100$, $200x200$, $300x300$ e $400x400$, a média da soma do tamanho das duas listas de ancestrais foram, respectivamente, de 39,05, 58,39, 75,39 e 89,27, enquanto que os números médios de comparações para a identificação do ancestral comum de maior profundidade foram de 5,89, 6,89, 7,91 e 8,14.

Diante desses valores, observa-se que além de o processo de identificação do $\theta - loop$ ser linear quando utilizada a estrutura em árvore, apenas parte dos nós são necessários de serem verificados.

5.2 Comparação entre os recálculos dos custos atualizados para problemas com variáveis do tipo inteira

A vantagem de tempo de resolução apresentada descrita na seção 5.1 é resultado de artifícios de programação, sendo que a estrutura em árvore permite que o $\theta - loop$ seja encontrado com menos número de operações.

Na presente seção são comparadas, para a implementação em árvore, a diferença de tempo de resolução entre a cada iteração recalcular todos os custos atualizados pelo método MODI e recalcular apenas os necessários pela expressão (54), ou seja, é analisada a vantagem da utilização de um resultado matemático teórico.

Com a utilização da metodologia de cálculo dos novos valores dos custos atualizados demonstrada no capítulo 3, não ocorre diminuição no número de iterações para obtenção da solução ótima, apenas redução no número de operações para recálculo dos custos atualizados.

Os problemas utilizados, assim como as soluções iniciais, para a presente análise foram os mesmos utilizados para os resultados apresentados na seção 5.1. Como o recálculo dos custos atualizados pela expressão (54) é baseado na utilização da estrutura em árvore, na presente seção não será utilizada a estrutura em quadro para comparação de tempos.

No presente capítulo os resultados referentes à implementação em árvore para o recálculo dos custos atualizados pela expressão (54) serão referenciados por A_Otimizado.

5.2.1 Comparação entre o recálculo dos custos atualizados para o Problema de Transporte para variáveis do tipo inteira

Na tabela 3 é apresentado o tempo médio de resolução, em milissegundos, de 100 problemas para cada configuração de parâmetros. A situação de recálculo dos custos atualizados pelo método MODI em árvore é chamada de A_MODI enquanto o recálculo pela expressão (54) é chamado de A_Otimizado.

TABELA 3 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO PARA A_MODI E A_Otimizado

Dimensão	Custo Máximo	Tempo médio de resolução A_MODI (ms)	Tempo médio de resolução A_Otimizado (ms)	Economia de A_Otimizado em relação à A_MODI	Número médio de iterações
100	10	48,29	9,95	79,40%	88,32
100	100	112,55	22,91	79,64%	204,61
100	1000	114,41	23,31	79,63%	207,54
100	10000	113,61	23,03	79,73%	206,08
200	10	248,26	46,14	81,41%	116,93
200	100	1.136,43	220,79	80,57%	533,75
200	1000	1.127,02	218,11	80,65%	529,27
200	10000	1.129,31	218,62	80,64%	530,34
300	10	640,77	111,17	82,65%	136,05
300	100	4.314,78	810,42	81,22%	908,32
300	1000	4.338,37	809,94	81,33%	912,91
300	10000	4.292,83	805,81	81,23%	903,25
400	10	1.311,58	226,26	82,75%	156,77
400	100	9.917,43	1.957,75	80,26%	1.171,88
400	1000	11.220,34	2.184,67	80,53%	1.323,63
400	10000	11.288,12	2.157,72	80,89%	1.331,85

FONTE: O Autor (2014)

A redução do tempo médio de resolução quando utilizada a expressão (54) para o recálculo dos custos atualizados foi de 79,40% a 82,75% em relação ao método MODI também implementado em estrutura de árvore. Considerando todos os exemplos testados a redução média foi de 80,78% no tempo de resolução.

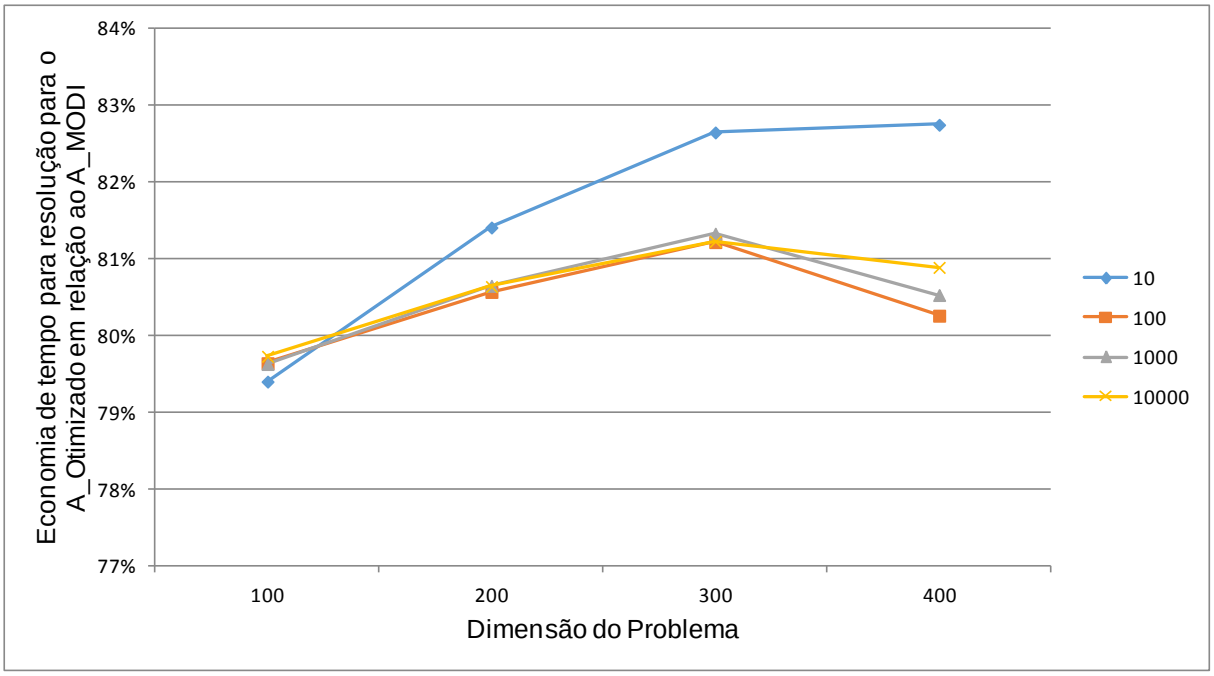


GRÁFICO 4 – ECONOMIA DE TEMPO DE RESOLUÇÃO PARA PROBLEMAS DE TRANSPORTE DO A_Otimizado EM RELAÇÃO AO A_MODI

FONTE: O Autor (2014)

A economia média de tempo gerada pela utilização do procedimento MudaArvoreOtimizada em relação à MudaArvore foi maior para os caso em que o número de dígitos de precisão da matriz de custos é menor (valores de 1 a 10). Já para os outros intervalos testados, não observa-se diferença significativa (gráfico 4)

Para precisões de custos de 2 a 4 dígitos, observa-se (gráfico 4) que houve decréscimo do percentual de economia quando a dimensão do problema aumentou de 300 para 400, entretanto, não é possível realizar qualquer tipo de conclusão de que o comportamento será mantido para problemas maiores.

5.2.2 Comparação entre o recálculo dos custos atualizados para o Problema de Designação para variáveis do tipo inteira

Na tabela 4 é apresentado o tempo médio de resolução, em milissegundos, de 100 problemas para cada configuração de parâmetros. A situação de recálculo dos custos atualizados pelo método MODI em árvore é chamada de A_MODI enquanto o recálculo pela expressão (54) é chamado de A_Otimizado.

TABELA 4 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO ENTRE MudaArvore e MudaArvoreOtimizada PARA O PROBLEMA DE DESIGNAÇÃO

Dimensão	Custo Máximo	Tempo médio de resolução A_MODI (ms)	Tempo médio de resolução A_Otimizado (ms)	Economia de A_Otimizado em relação à A_MODI	Número médio de iterações
100	10	78,03	8,06	89,67%	149,37
100	100	179,16	19,94	88,87%	340,04
100	1000	179,36	19,94	88,88%	340,56
100	10000	180,96	20,22	88,83%	343,05
200	10	486,66	44,64	90,83%	244,50
200	100	1.792,93	171,11	90,46%	880,46
200	1000	1.793,87	171,33	90,45%	880,81
200	10000	1.800,37	172,31	90,43%	884,40
300	10	1.532,43	125,25	91,83%	348,66
300	100	6.752,16	583,06	91,36%	1.484,90
300	1000	6.880,95	594,6	91,36%	1.511,97
300	10000	6.908,53	597,85	91,35%	1.517,70
400	10	3.574,35	283,19	92,08%	460,86
400	100	15.317,82	1267,7	91,72%	1.897,79
400	1000	18.135,97	1513,79	91,65%	2.239,24
400	10000	18.242,47	1524,28	91,64%	2.249,93

FONTE: O Autor (2014)

A redução do tempo médio de resolução para Problemas de Designação quando utilizada a expressão (54) para o recálculo dos custos atualizados foi de 88,83% a 92,08% em relação ao método MODI também implementado em estrutura de árvore. Considerando todos os exemplos testados a redução média foi de 90,71% no tempo de resolução.

Em termos percentual de economia, a utilização do procedimento MudaArvoreOtimizada no lugar do procedimento MudaArvore trouxe mais vantagem ainda para os problemas de designação do que para os Problemas de Transporte.

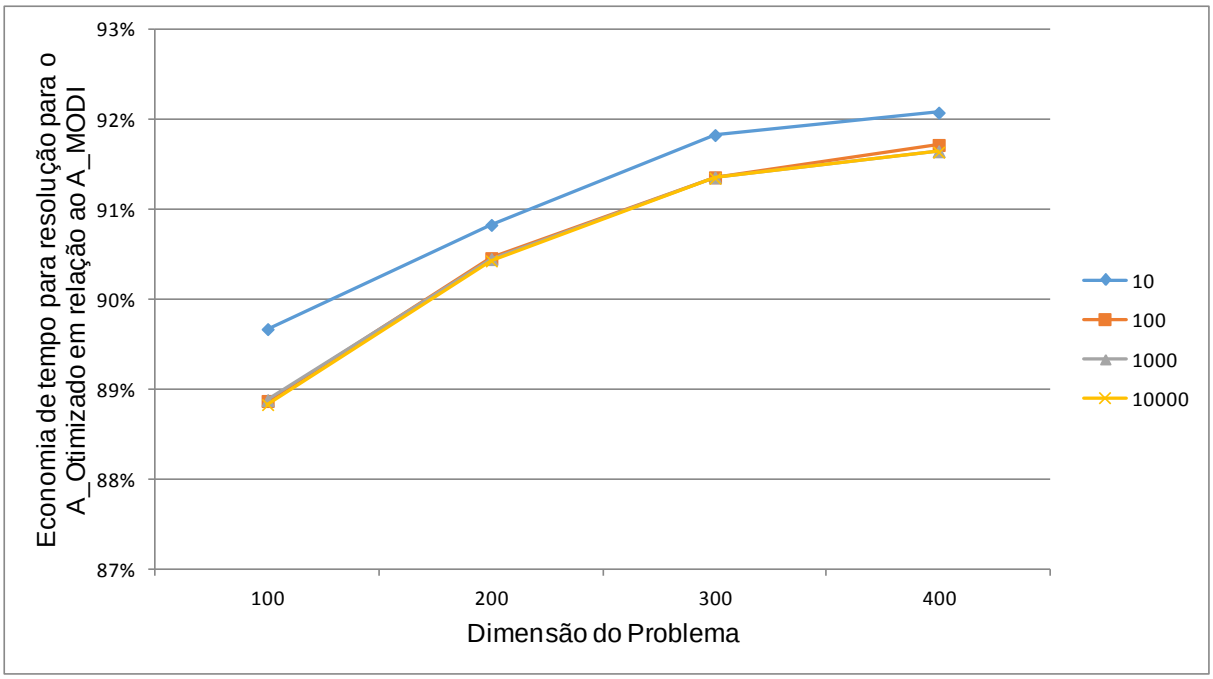


GRÁFICO 5 – ECONOMIA DE TEMPO DE RESOLUÇÃO PARA PROBLEMAS DE DESIGNAÇÃO DO A_Otimizado EM RELAÇÃO AO A_MODI

FONTE: O Autor (2014)

Para a resolução de Problemas de Designação, a economia média de tempo gerada pela utilização do procedimento MudaArvoreOtimizada em relação à MudaArvore, assim como para o caso em que as ofertas e demandas eram valores entre 300 e 800, foi maior para os caso em que o número de dígitos de precisão da matriz de custos é menor (valores de 1 a 10).

Ao contrário do que acontecia para Problemas de Transporte, no caso dos Problemas de Designação observou-se (gráfico 5) que o percentual de economia cresceu em função do crescimento do tamanho do problema. Entretanto, vale

ressaltar que não existe garantia alguma de que o comportamento continuará para problemas de dimensões maiores do que 400.

Ao serem analisados os resultados das tabelas 1, 2, 3 e 4, constata-se que a utilização da expressão (54) para o recálculo dos custos atualizados em conjunto com a estrutura de árvore gerou uma economia média de tempo de 93,03% para resolução dos Problemas de Transporte testados com ofertas e demandas entre 300 e 800. Quando a mesma comparação é utilizada para problemas com ofertas e demandas igual a 1 (Problemas de Designação) a economia é de 97,67%.

Desta forma, com a utilização de uma estrutura que permitiu encontrar o $\theta - loop$ de forma mais rápida e um resultado matemático teórico para recálculo dos custos atualizados, foi possível uma redução significativa no tempo de resolução em relação à implementação tradicional em quadro.

5.3 COMPARAÇÃO ENTRE OS RECÁLCULOS DOS CUSTOS ATUALIZADOS PARA PROBLEMAS COM VARIÁVEIS DO TIPO PONTO FLUTUANTE DE SIMPLES PRECISÃO

Os exemplos utilizados nas seções 5.1 e 5.2 foram gerados aleatoriamente com valores sempre inteiros. Na seção 5.3 são apresentados testes realizados com variável do tipo ponto flutuante de simples precisão (*single* na linguagem VB.NET), aumentando assim o número de valores possíveis que custos, ofertas e demandas podem assumir.

5.3.1 Comparação entre o recálculo dos custos atualizados para o Problema de Transporte para variáveis do tipo ponto flutuante de simples precisão

Na tabela 5 é apresentado o tempo médio de resolução, em milissegundos, de 100 problemas de cada tamanho. A situação de recálculo dos custos atualizados pelo método MODI em árvore é, assim como na seção 5.2, chamada de A_MODI enquanto o recálculo pela expressão (54) é chamado de A_Otimizado.

TABELA 5 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO PARA A_Otimizado e A_MODI

Dimensão	Tempo médio de resolução A_MODI (ms)	Tempo médio de resolução A_Otimizado (ms)	Economia de tempo da resolução de A_Otimizado em relação à A_MODI	Número médio de iterações
100	118,33	25,57	78,39%	214,96
200	1.177,61	238,60	79,74%	549,73
300	4.477,00	868,59	80,60%	933,35
400	11.963,68	2.276,53	80,97%	1.396,66

FONTE: O Autor (2014)

Analisando os tempos médios de resolução e comparando os resultados apresentados nas tabelas 3 e 5, verifica-se que ocorre um aumento de tempo com a utilização de variáveis do tipo ponto flutuante de simples precisão em relação à utilização de variáveis do tipo inteiro.

Quando utilizadas variáveis do tipo ponto flutuante de simples precisão, a redução do tempo médio de resolução quando utilizada a expressão (54) para o recálculo dos custos atualizados foi de 78,39% a 80,97% em relação ao método MODI também implementado em estrutura de árvore. Considerando todos os exemplos testados a redução média foi de 79,92% no tempo de resolução.

Os valores de percentuais de economia de tempo são similares tanto para os exemplos com variáveis inteiras (com exceção de quando os custos possuíam apenas um dígito de precisão) quanto para os com variáveis do tipo ponto flutuante de simples precisão. Ou seja, os ganhos percentuais em tempo de resolução pela utilização do procedimento MudaArvoreOtimizada em relação ao MudaArvore não possuem influência significativa do tipo da variável, para a implementação realizada no presente trabalho, para o Problema de Transporte.

5.3.2 Comparação entre o recálculo dos custos atualizados para o Problema de Designação para variáveis do tipo ponto flutuante de simples precisão

Na tabela 6 é apresentado o tempo médio de resolução, em milissegundos, de 100 problemas de cada tamanho.

TABELA 6 – COMPARAÇÃO DE TEMPOS DE RESOLUÇÃO PARA AS ESTRUTURAS EM QUADRO E EM ÁRVORE

Dimensão	Tempo médio de resolução A_MODI (ms)	Tempo médio de resolução A_Otimizado (ms)	Economia de tempo da resolução de A_Otimizado em relação à A_MODI	Número médio de iterações
100	187,53	25,78	86,25%	344,78
200	1.851,39	228,97	87,63%	886,73
300	7.097,57	826,46	88,36%	1523,00
400	18.740,50	2.073,32	88,94%	2.260,60

FONTE: O Autor (2014)

A redução do tempo médio de resolução para Problemas de Designação quando utilizada a expressão (54) para o recálculo dos custos atualizados foi de 86,25% a 88,84% em relação ao método MODI também implementado em estrutura de árvore. Considerando todos os exemplos testados a redução média foi de 87,79% no tempo de resolução.

Embora o percentual de economia de tempo gerado pelo procedimento MudaArvoreOtimizada em relação ao MudaArvore tenha sido, para o caso de variáveis do tipo ponto flutuante de simples precisão, menor do que a redução média de 90,71% para variáveis do tipo inteiro, novamente observa-se que o ganho percentual em tempo do procedimento MudaArvoreOtimizada foi maior para Problemas de Designação em relação à Problemas de Transporte.

5.3.3 Análise sobre o número de recálculos de custos atualizados para A_Otimizado

A única diferença entre o A_MODI e o A_Otimizado está no recálculo dos custos atualizados, enquanto no primeiro todos os custos atualizados são recalculados com base nos valores das variáveis duais, no outro apenas parte deles são recalculados com base na expressão (54). Nas tabelas 7 e 8 são apresentados para o método A_Otimizado o número médio, por iteração, de custos atualizados recalculados para os mesmos exemplos das seções 5.3.1 e 5.3.2, respectivamente.

TABELA 7 – QUANTIDADE E PERCENTUAL DE CUSTOS ATUALIZADOS RECALCULADOS PARA OS PROBLEMAS DE TRANSPORTE

Dimensão	Número médio, por iteração, de custos atualizados recalculados	Percentual médio, por iteração, de custos atualizados recalculados
100	2.852,07	29,10%
200	11.616,53	29,33%
300	26.347,76	29,47%
400	47.203,58	29,65%

FONTE: O Autor (2014)

Com base nas informações da tabela 7, observa-se que em A_Otimizado menos de 30% dos custos atualizados são, em média, recalculados a cada iteração. Além disso, como no método A_MODI são realizadas duas operações para cada custo atualizado recalculado enquanto no método A_Otimizado apenas uma, a economia de tempo fica maior ainda e são conseguidos os resultados apresentados na tabela 3.

TABELA 8 – QUANTIDADE E PERCENTUAL DE CUSTOS ATUALIZADOS RECALCULADOS PARA OS PROBLEMAS DE DESIGNAÇÃO

Dimensão	Número médio, por iteração, de custos atualizados recalculados	Percentual médio, por iteração, de custos atualizados recalculados
100	685,26	6,99%
200	1.900,52	4,80%
300	3.442,79	3,85%
400	5.261,61	3,31%

FONTE: O Autor (2014)

Para os casos dos Problemas de Designação, um percentual de custos atualizados recalculados menor do que os observados para os Problemas de Transporte foi constatado. Além disso, o percentual de recálculos diminuiu em função do crescimento da dimensão do problema (tabela 8). Estes explicam os resultados da tabela 4.

O percentual de custos atualizados que precisam ser recalculados no método A_Otimizado depende de quantos nós de origem e destino pertencem a

árvore $\mathcal{H}$ a cada iteração. Diante disso, na tabela 9 são apresentadas as quantidades médias de origens e destinos pertencentes a $\mathcal{H}$, a cada iteração, para as diferentes dimensões dos Problemas de Transporte e Designação testados.

TABELA 9 – NÚMERO MÉDIO POR ITERAÇÃO DE ORIGENS E DESTINOS EM $\mathcal{H}$

Dimensão	Problemas de Transporte		Problemas de Designação	
	Média, por iteração, de Origens em $\mathcal{H}$	Média, por iteração, Destinos em $\mathcal{H}$	Média, por iteração, Origens em $\mathcal{H}$	Média, por iteração, Destinos em $\mathcal{H}$
100	28,41	28,44	5,12	5,15
200	57,82	57,97	6,83	6,85
300	87,09	87,07	8,13	8,14
400	117,87	118,07	9,14	9,15

FONTE: O Autor (2014)

O fato de as árvores dos Problemas de Designação possuírem menos elementos do que as dos Problemas de Transporte (tabela 9) faz com que menos custos atualizados sejam necessários serem recalculados e, por isso, ocorre maior economia de tempo quando utilizado o A_Otimizado para este tipo de problema.

5.3.4 Análise sobre o $\theta - loop$

O número de operações necessárias para encontrar o $\theta - loop$, quando a implementação é realizada em árvore, depende do número de elementos nas listas AncestraisOirgem e AncenstraisDestino e do valor de PontoEncontro, que representa o número de comparações que são necessárias para encontrar o ancestral comum de maior profundidade entre o nó origem e o nó destino referentes à variável que está entrando na base.

TABELA 10 – NÚMERO MÉDIO DE ELEMENTOS NAS LISTAS AncestraisOrigem E AncestraisDestino E VALOR MÉDIO DE PontoEncontro PARA OS PROBLEMAS DA SEÇÃO 5.3

Dimensão do Problema	Problemas de Transporte			Problemas de Designação		
	Número de elementos		Ponto de Encontro	Número de elementos		Ponto de Encontro
	Ancestrais Origem	Ancestrais Destino		Ancestrais Origem	Ancestrais Destino	
100	18,14	18,12	5,51	19,72	18,49	5,33
200	28,65	28,66	7,22	29,91	28,36	6,96
300	37,01	36,91	8,01	37,76	37,35	7,90
400	44,31	44,59	8,56	45,22	44,70	8,65

FONTE: O Autor (2014)

Para os casos de Problema de Transporte e Designação, o número de elementos das listas AncestraisOrigem e AncestraisDestino e a variável PontoEncontro foram similares. Além disso, a razão do crescimento dos valores destas variáveis foi menor do que um.

Sobre os comprimentos médios do $\theta - loop$ para os Problemas de Transporte (e Designação) de tamanhos $100x100$, $200x200$, $300x300$ e $400x400$, eles foram, respectivamente, 26,23 (28,54), 43,86 (45,35), 58,90 (60,32) e 72,77 (73,62). Diante disto, tem-se que o comprimento do $\theta - loop$ também tem crescimento abaixo do linear em função do aumento da dimensão do problema.

O comprimento médio do $\theta - loop$ seria o mesmo para o caso da implementação em estrutura de quadro. A utilização da estrutura de árvore não altera o $\theta - loop$, apenas faz com que ele seja encontrado com menor esforço computacional.

5.4 DESEMPENHO PARA A UTILIZAÇÃO DA LISTA *ReferenciaCAN*

A inclusão da lista *ReferenciaCAN*, explicada na seção 4.3, foi utilizada com o objetivo de tentar diminuir o número de comparações para a escolha da variável não básica a entrar na base. Com isso, existia a expectativa de que ocorreria um

maior número de iterações para a resolução do problema, mas o tempo gasto em cada iteração poderia ser menor e compensar.

Os mesmos problemas de variável do tipo ponto flutuante de simples precisão utilizados para os resultados da seção 5.3 foram então resolvidos pelo procedimento ResolveProblemaTransporteListaCAN e os resultados são apresentados nas seções 5.4.1 e 5.4.2.

5.4.1 Desempenho para a utilização da lista *ReferenciaCAN* para Problemas de Transporte

Assim como para os casos anteriores, são apresentado inicialmente os resultados obtidos para Problemas de Transporte. Na tabela 11 são descritos o número médio de iterações e o tempo médio de resolução para comparação do efeito da utilização da lista *ReferenciaCAN*.

TABELA 11 – COMPARAÇÃO DE TEMPOS E ITERAÇÕES PARA O PROBLEMA DE TRANSPORTE GERADAS PELA UTILIZAÇÃO DA LISTA *ReferenciaCAN*

Tamanho	Tempo médio de resolução (ms)		Economia	Número médio de iterações		Acréscimo
	A_Otimizada	ReferenciaCAN		A_Otimizada	ReferenciaCAN	
100	26,38	20,35	22,86%	214,96	291,46	35,59%
200	240,59	187,39	22,11%	549,73	780,29	41,94%
300	871,45	683,08	21,62%	933,37	1.369,87	46,77%
400	2.300,93	1.763,95	23,34%	1.397,26	2.057,87	47,28%

FONTE: O Autor (2014)

Assim como esperado, ocorreu um aumento do número de iterações quando a lista *ReferenciaCAN* foi utilizada. Entretanto, o decréscimo ocorrido no tempo de cada iteração fez com que o tempo final de resolução fosse, em média, 22,48% menor do que quando é realizada a busca da variável não básica com o custo atualizado mais negativo entre todas as não básicas.

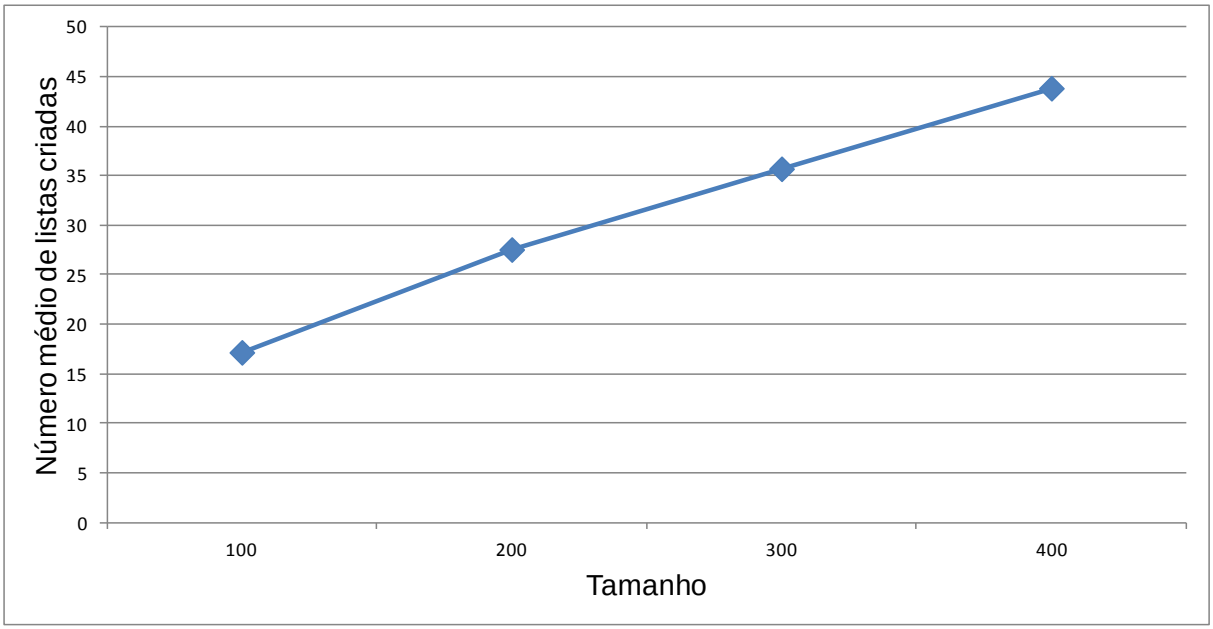


GRÁFICO 6 – NÚMERO DE VEZES QUE A LISTA FOI REINICIALIZADA PARA A RESOLUÇÃO DO PROBLEMA DE TRANSPORTE

FONTE: O Autor(2014)

O número de vezes que a lista é reiniciada (gráfico 6), assim como era esperado, cresce em função do aumento da dimensão do problema. Porém, o percentual de economia de tempo (tabela 11) não sofre tanta influência da dimensão do problema.

5.4.2 Desempenho para a utilização da lista *ReferenciaCAN* para Problemas de Designação

Seguindo o mesmo raciocínio das seções anteriores, aqui são apresentados na tabela 12, para os Problemas de Designação, os resultados de tempo médio de resolução e número médio de iterações quando utilizada a lista *ReferenciaCAN* e, para comparação, os obtidos anteriormente por A_Otimizada.

TABELA 12 – COMPARAÇÃO DE TEMPOS E ITERAÇÕES PARA O PROBLEMA DE DESIGNAÇÃO GERADAS PELA UTILIZAÇÃO DA LISTA *ReferenciaCAN*

Tamanho	Tempo médio de resolução (ms)		Economia	Número médio de iterações		Acréscimo
	A_Otimizada	ReferenciaCAN		A_Otimizada	ReferenciaCAN	
100	25,78	17,68	31,42%	344,78	519,25	50,60%
200	228,97	122,36	46,56%	886,73	1.404,49	58,39%
300	826,46	372,73	54,90%	1.523,02	2.500,40	64,17%
400	2.073,32	865,63	58,25%	2.259,39	3.753,59	66,13%

FONTE: O Autor (2014)

Para o caso do Problema de Designação, quando utilizada a lista *ReferenciaCAN*, ocorreu tanto um aumento no número de iterações como no percentual de economia de tempo maior do que os observados para o caso do Problema de Transporte.

O percentual de economia, que no caso dos Problemas de Transporte não era influenciado (considerando as dimensões estudadas), foi maior para problemas de dimensões maiores para o caso dos Problemas de Designação (tabela 11 e gráfico 7).

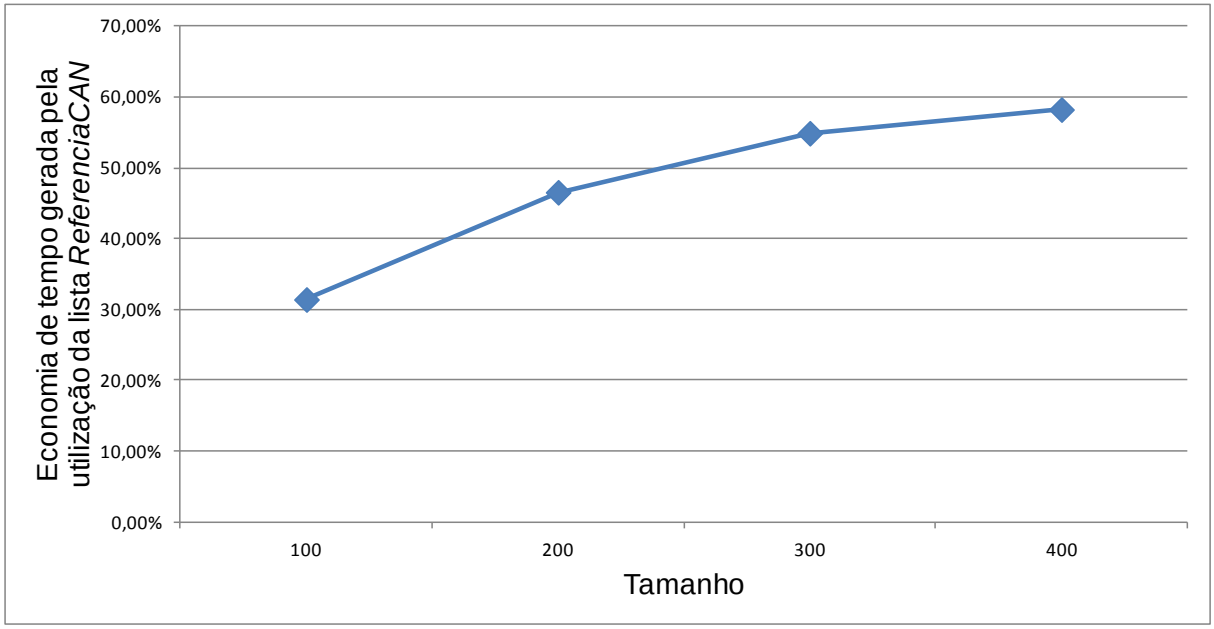


GRÁFICO 7 – ECONOMIA DO TEMPO DE RESOLUÇÃO GERADA PELA UTILIZAÇÃO DA LISTA *ReferenciaCAN* NOS PROBLEMAS DE DESIGNAÇÃO

FONTE: O Autor(2014)

O número de vezes que a lista *ReferenciaCAN* é reiniciada para os problemas de designação (gráfico 8), assim como para o caso dos Problemas de

Transporte, cresce em função do aumento da dimensão do problema, porém com uma razão menor.

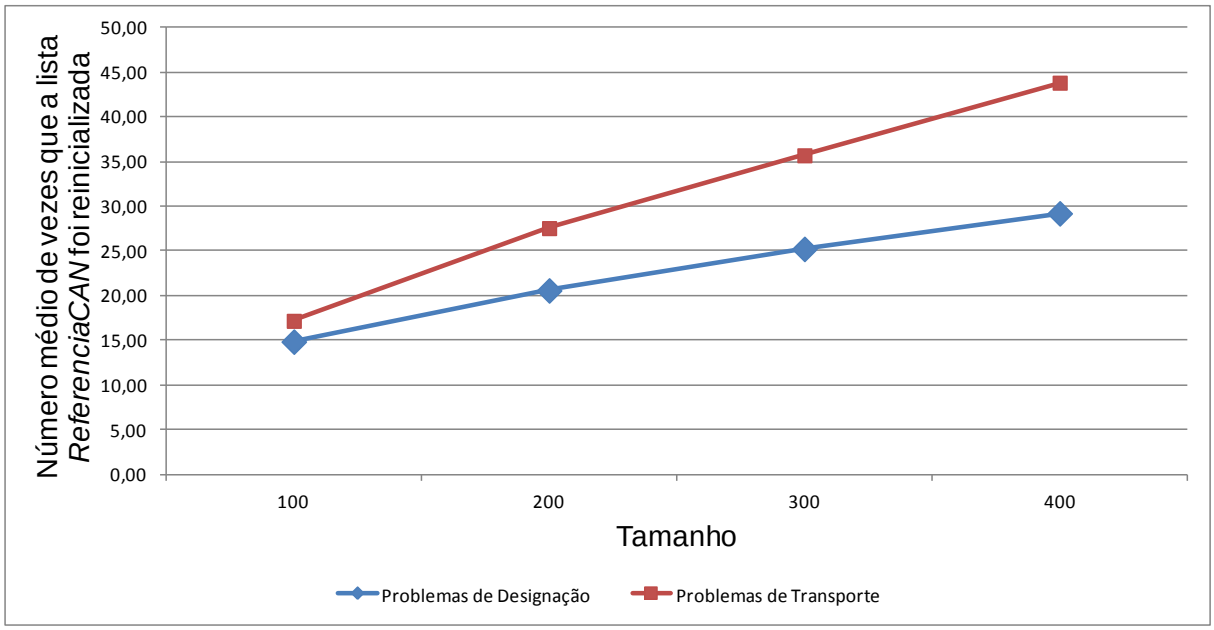


GRÁFICO 8 – COMPARAÇÃO DO NÚMERO DE VEZES QUE A LISTA ReferenciaCAN É ATUALIZADA PARA PROBLEMAS DE TRANSPORTE E DESIGNAÇÃO

FONTE: O Autor (2014)

Diante dos resultados observados pela utilização da lista *ReferenciaCAN*, observou-se a validade da utilização dela. Para tentar melhorar ainda mais o tempo, de resolução foi incluído o parâmetro *PercentualEconomiaAnterior*, a fim de fazer com que quando a máxima economia unitária das variáveis não básicas referenciadas na lista *ReferenciaCAN* seja inferior a economia unitária máxima permitida em relação à iteração anterior, a lista *ReferenciaCAN* seja reinicializada. Os resultados obtidos para os testes de diferentes valores do parâmetro são apresentados na seção 5.5

5.5 INCLUSÃO DO PARÂMETRO *PercentualEconomiaAnterior*

O valor do parâmetro *PercentualEconomiaAnterior* interfere diretamente no número de vezes que a lista *ReferenciaCAN* é reinicializada. Caso ele seja 0, seria a mesma coisa que não existisse e caso seja 1 seria o equivalente à lista

ReferenciaCAN não existir. Nas seções 5.5.1 e 5.5.2 são apresentados os resultados obtidos para valores do parâmetro no intervalo de 0 a 1.

Para definir bons valores para parâmetro *PercentualEconomiaAnterior*, foram realizados testes para os mesmos problemas utilizados para os resultados das seções 5.3 e 5.4. Além disso, foi utilizado o procedimento MudaArvoreOtimizada para o recálculo dos custos atualizados.

5.5.1 Resultados para o parâmetro *PercentualEconomiaAnterior* nos Problemas de Transporte

Os resultados obtidos para diferentes valores de *PercentualEconomiaAnterior* para os Problemas de Transporte são apresentados na tabela 13.

TABELA 13 – TEMPO DE RESOLUÇÃO PARA PROBLEMAS DE TRANSPORTE UTILIZANDO O PARÂMETRO *PercentualEconomiaAnterior*

PercentualEconomiaAnterior	Tempo de resolução (em ms) para diferentes tamanhos de problemas			
	100	200	300	400
0,00	20,35	187,39	683,08	1.763,95
0,10	20,24	186,72	679,92	1.753,16
0,20	19,88	185,49	680,31	1.751,11
0,30	19,83	183,86	670,64	1.729,38
0,40	19,48	181,88	665,26	1.706,70
0,50	18,58	174,60	640,49	1.644,34
0,60	17,92	162,44	577,74	1.514,28
0,70	17,40	153,75	544,26	1.377,32
0,80	18,59	164,39	575,21	1.436,63
0,90	21,25	192,18	685,11	1.742,38
1,00	26,38	240,59	871,45	2.300,93

FONTE: O Autor (2014)

Os valores da tabela 13 são também apresentados no gráfico 9, sendo possível estimar que o parâmetro ótimo, com base nos exemplos testados, deve estar no intervalo de 0,6 a 0,8.

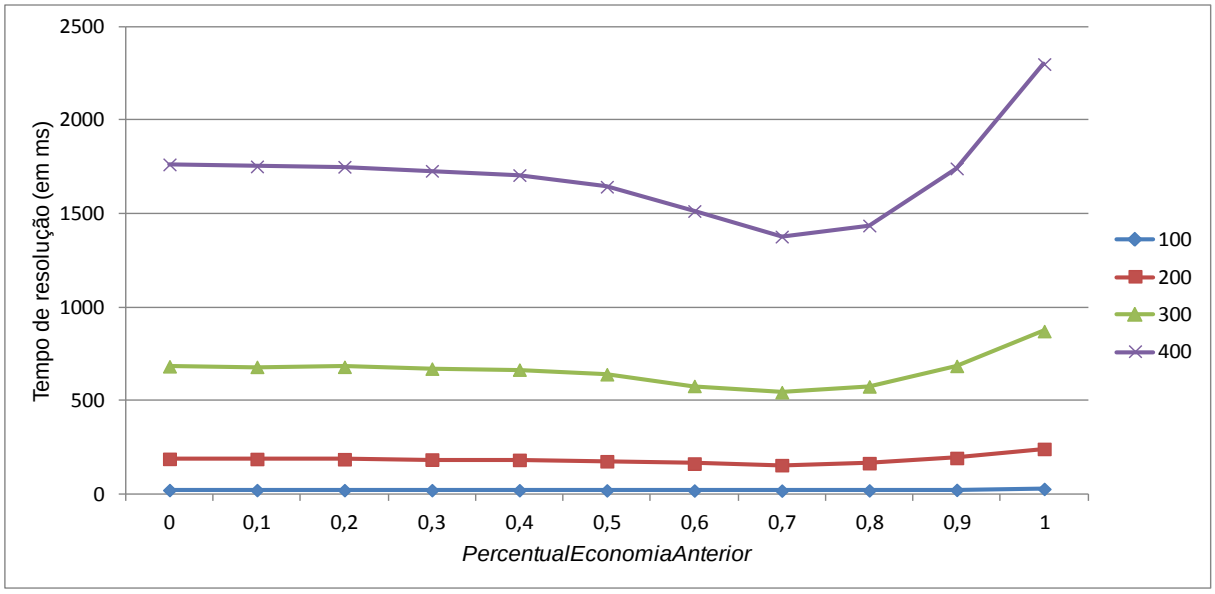


GRÁFICO 9 – TEMPO DE RESOLUÇÃO DOS PROBLEMAS DE TRANSPORTE PARA DIFERENTES VALORES DE *PercentualEconomiaAnterior*

FONTE: O Autor (2014)

Como existe diferença nas ordens de grandeza dos tempos de resolução para diferentes dimensões de problemas, no gráfico 10 são apresentados os percentuais de economia de tempo gerados para diferentes valores de *PercentualEconomiaAnterior* em relação à situação em que *PercentualEconomiaAnterior*=1, lembrando que este parâmetro ser igual a 1 é equivalente à lista *ReferenciaCAN* não ser utilizada.

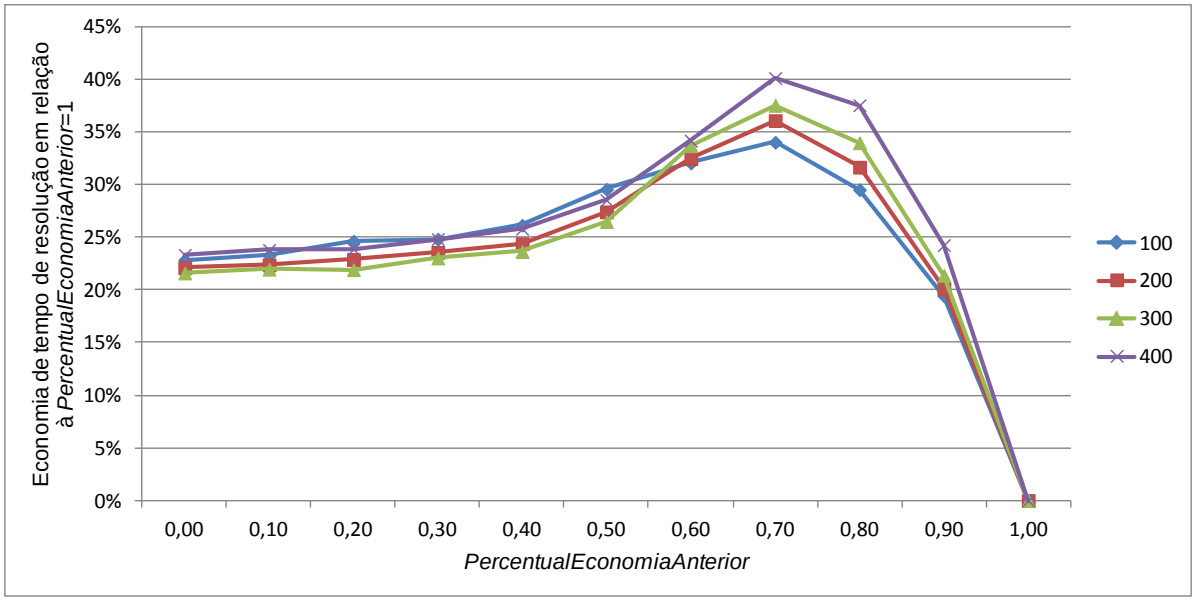


GRÁFICO 10 – ECONOMIA DE TEMPO DE RESOLUÇÃO EM RELAÇÃO À *PercentualEconomiaAnterior*=1

FONTE: O Autor (2014)

Com a inclusão do parâmetro *PercentualEconomiaAnterior* com o valor de 0,70, a redução em tempo computacional para resolução dos Problemas de Transporte, que era de 21,62% a 23,34% (pela utilização do procedimento MudaArvoreOtimizada em relação ao MudaArvore), passa a ser de 34,04% a 40,14% quando além de utilizar o procedimento são utilizadas a lista *ReferenciaCAN* e o parâmetro *PercentualEconomiaAnterior*=0,70.

Se a comparação for realizada em relação ao tempo de resolução com a lista *ReferenciaCAN* sem o parâmetro *PercentualEconomiaAnterior*, a economia de tempo gerada pela inclusão de *PercentualEconomiaAnterior* =0,7 fica entre 14,50% e 21,92%.

Com os resultados obtidos para os diferentes valores do parâmetro *PercentualEconomiaAnterior*, estima-se que o melhor valor para ele esteja entre 0,60 e 0,80 e, por isso, testes com maior precisão precisariam ser realizados neste intervalo. Para isso, foram realizados inicialmente testes no intervalo de 0,71 a 0,79 e constatou-se (resultados apresentados na tabela 14) que os menores tempos ocorreram para *PercentualEconomiaAnterior*=0,79.

TABELA 14 – TEMPO DE RESOLUÇÃO PARA PROBLEMAS DE TRANSPORTE UTILIZANDO O PARÂMETRO PercentualEconomiaAnterior

PercentualEconomia Anterior	Tempo de resolução (em ms) para diferentes tamanhos de problemas			
	100	200	300	400
0,70	17,40	153,75	544,26	1.377,32
0,71	18,22	156,15	548,66	1.379,74
0,72	17,79	155,29	539,33	1.369,16
0,73	17,51	154,65	534,59	1.336,38
0,74	17,33	153,72	534,23	1.323,97
0,75	17,47	152,73	527,87	1.308,77
0,76	16,87	146,96	509,32	1.280,16
0,77	16,57	146,01	507,97	1.276,94
0,78	16,50	145,26	508,19	1.284,00
0,79	17,08	147,11	507,39	1.265,09
0,80	18,59	164,39	575,21	1.436,63

FONTE: O Autor (2014)

Os tempos de resolução para *PercentualEconomiaAnterior* entre 0,60 e 0,70 foram maiores dos que quando utilizado o parâmetro como sendo 0,70 e são aqui omitidos. Para analisar o desempenho de *PercentualEconomiaAnterior* =0,79, na

seção 5.7.1 são apresentados os resultados de testes realizados para problemas de dimensões maiores.

5.5.2 Resultados para o parâmetro *PercentualEconomiaAnterior* nos Problemas de Designação

As mesmas análises que foram realizadas para os Problemas de Transporte descritas na seção 5.5.1 também foram realizadas para os Problemas de Designação e os resultados são resumidos na tabela 15.

TABELA 15 – TEMPO DE RESOLUÇÃO PARA PROBLEMAS DE DESIGNAÇÃO UTILIZANDO O PARÂMETRO PercentualEconomiaAnterior

PercentualEconomia Anterior	Tempo de resolução (em ms) para diferentes dimensões de problemas			
	100	200	300	400
0,00	17,68	122,36	372,73	865,63
0,10	17,45	121,54	369,36	854,66
0,20	17,11	121,56	370,75	850,69
0,30	17,06	121,33	371,98	844,78
0,40	16,65	121,45	367,25	842,40
0,50	16,32	113,13	348,81	811,53
0,60	15,01	102,47	309,01	697,56
0,70	13,75	91,05	264,60	600,35
0,71	14,37	92,24	265,96	600,87
0,72	14,05	91,91	261,61	595,88
0,73	13,80	91,46	259,21	581,70
0,74	13,67	91,02	258,92	575,65
0,75	13,80	90,20	256,17	570,30
0,76	13,30	86,74	247,29	556,37
0,77	13,09	86,31	246,54	556,14
0,78	13,02	85,87	246,63	558,17
0,79	13,47	86,85	246,10	550,35
0,80	13,40	87,26	246,07	552,40
0,90	15,38	101,74	293,21	655,75
1,00	25,78	228,97	826,46	2.073,32

FONTE: O Autor (2014)

Para os Problemas de Designação, os melhores valores do parâmetro *PercentualEconomiaAnterior* foram no intervalo de 0,78 a 0,80. Como para os Problemas de Transporte o melhor valor já era de 0,79, foi escolhido para ser utilizado nos problemas das seções 5.7 e 5.8 o valor de 0,79.

5.6 ANÁLISE PARA PROBLEMAS DE DIMENSÕES MAIORES

Até a seção 5.5 foram descritos os resultados para as implementações realizadas, utilizando exemplo de dimensão 100 a 400. Para analisar o desempenho dos procedimentos que permitiram os melhores resultados para as seções anteriores, a implementação em árvore realizando o cálculo dos novos valores de custos atualizados pela expressão (54), com a inclusão da lista *ReferenciaCAN* e do parâmetro *PercentualEconomiaAnterior* com o valor de 0,79, foi utilizada para problemas de 800x800 e 1.600x1.600.

Para Problemas de Transporte de 800x800 o tempo médio de resolução foi de 17.234,91 milissegundos quando utilizada a configuração descrita no parágrafo anterior, já para quando foi utilizado o A_Otimizado, o tempo médio foi de 27.460,70 milissegundos. Assim, a redução média de tempo foi de 37,24%.

Nos Problemas de Transporte de tamanho 1600x1600 o tempo médio de resolução para o A_Otimizado foi de 281.087,90 milissegundos, enquanto a utilização da lista *ReferenciaCAN* com o parâmetro *PercentualEconomiaAnterior* sendo igual a 0,79 gerou um tempo médio de resolução para os mesmos exemplos de 166.223,00 milissegundos, ou seja, uma redução de 40,86%.

Com isso, observou-se também ganhos de tempo computacional pela utilização da lista *ReferenciaCAN* e do parâmetro *PercentualEconomiaAnterior*=0,79 para problemas de maiores dimensões que sequer haviam sido utilizados para a definição do melhor valor de *PercentualEconomiaAnterior*.

5.7 ANÁLISE PARA PROBLEMAS DE DESIGNAÇÃO COM $m \gg n$

Como descrito anteriormente, os problemas de designação retangulares possuem utilidade como subproblemas em outros algoritmos. Diante disso, nesta seção são apresentados tempos de resolução para este tipo de problema quando

utilizado o algoritmo A_Otimizado e comparados com os obtidos quando utilizada a lista *ReferenciaCAN* e o parâmetro *PercentualEconomiaAnterior*=0,79.

Como problemas retangulares, para um mesmo número de variáveis, são mais rápidos de serem resolvidos do que problemas quadrados, foram utilizados para teste de problemas retangulares situações de 300.000 a 1.000.000 de variáveis e os resultados são apresentados na tabela 16.

TABELA 16 – DESEMPENHO DA UTILIZAÇÃO DA LISTA *ReferenciaCAN* E DO PARÂMETRO *PercentualEconomiaAnterior*=0,79 para PROBLEMAS DE DESIGNAÇÃO COM $m \gg n$

Número de Origens	Número de Destinos	Tempo (ms) A_Otimizado	Tempo (ms) Lista0,79	Economia Lista0,79
10.000	30	170,0	144,2	15,18%
10.000	40	292,2	208,0	28,82%
10.000	50	425,5	277,8	34,71%
10.000	60	636,0	397,0	37,58%
10.000	70	832,2	510,4	38,67%
10.000	80	1.158,7	675,6	41,69%
10.000	90	1.554,6	874,1	43,77%
10.000	100	1.925,7	1.072,5	44,31%

FONTE:O Autor (2014)

Para os problemas testados, observou-se que, fixado o número de origens, quanto maior o número de destinos, maior é a economia de tempo pela utilização do método EconomiaLista0,79 em relação ao A_Otimizado.

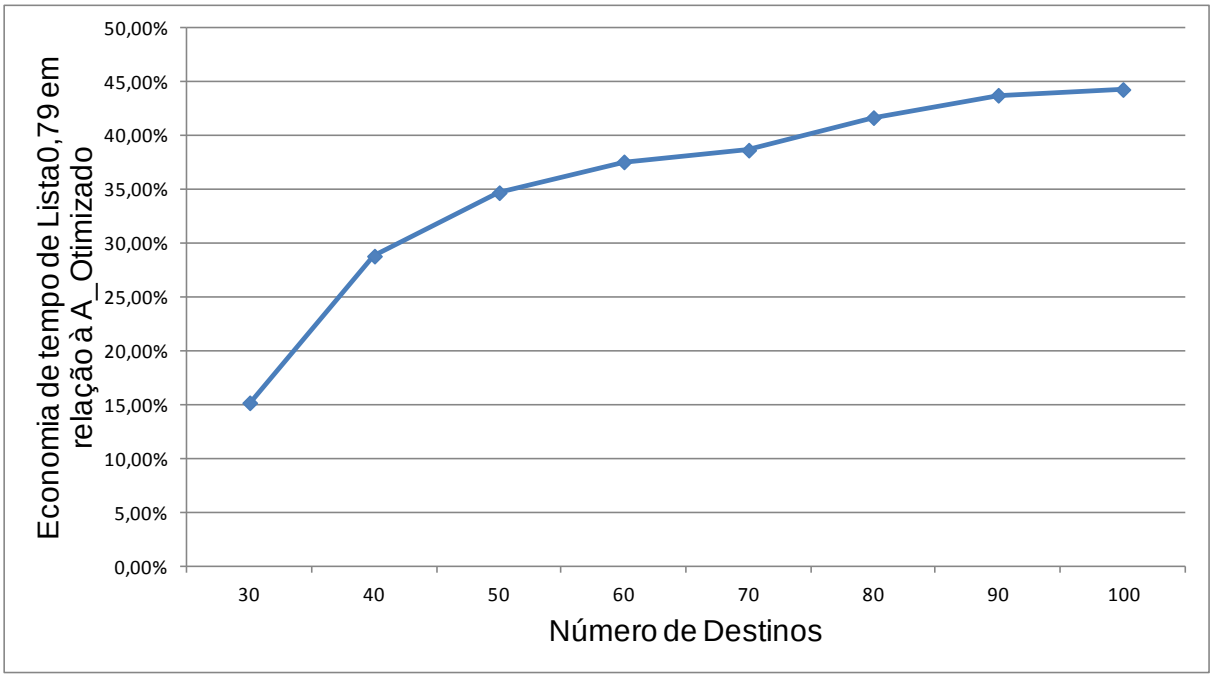


GRÁFICO 11 – ECONOMIA DE TEMPO DE Lista0,79 EM RELAÇÃO À A_Otimizado EM FUNÇÃO DO NÚMERO DE DESTINOS PARA 10.000 NÓS DE ORIGEM

FONTE: O Autor (2014)

Diante dos resultados obtidos para os problemas retangulares, constata-se para esta situação também o benefício, em termos de redução do tempo computacional de resolução, da utilização da lista *ReferenciaCAN* e do parâmetro *PercentualEconomiaAnterior*=0,79.

5.8 ANÁLISE FINAL PARA COMPARAÇÃO ENTRE A_MODI, A_Otimizado E LISTA *ReferenciaCAN* COM O PARÂMETRO *PercentualEconomiaAnterior*

Para realizar uma análise final sobre o ganho de tempo da utilização do A_Otimizado sobre o A_MODI e também do ganho adicional gerado pela inclusão da lista *ReferenciaCAN* e do parâmetro *PercentualEconomiaAnterior* foram realizados testes adicionais com número de origens e demandas variando de 20 a 360 (com intervalos de 20), considerando os problemas quadrados e retangulares. Parte destes resultados são aqui apresentados na tabela 17.

TABELA 17 – TEMPOS MÉDIOS DE RESOLUÇÃO EM MILISSEGUNDOS DE PROBLEMAS DE TRANSPORTE PELO MÉTODO A_MODI

Origens \ Destinos	120	160	200	240	280	320
120	205,84	420,37	676,90	939,53	1.152,69	1.386,68
160	405,84	546,42	947,47	1.497,21	2.063,05	2.550,00
200	678,42	956,74	1.177,61	1.851,47	2.781,00	3.861,42
240	954,90	1.468,00	1.797,95	2.161,16	3.071,58	4.555,37
280	1.167,58	2.215,84	2.783,47	3.135,21	3.544,89	5.144,53
320	1.424,05	2.577,21	3.733,84	4.638,00	4.917,89	5.479,26

FONTE: O Autor (2014)

Na tabela 18 são apresentadas, para os mesmos exemplos utilizados para obter os valores da tabela 17, as economias de tempo geradas pela utilização do procedimento MudaArvoreOtimizada (A_Otimizado) em relação ao uso do procedimento MudaArvore (A_MODI).

TABELA 18 – ECONOMIA DE TEMPO DE RESOLUÇÃO PELO MÉTODO A_Otimizado EM RELAÇÃO AO MÉTODO A_MODI

Origens \ Destinos	120	160	200	240	280	320
120	79,03%	81,96%	83,03%	83,35%	83,84%	83,84%
160	81,74%	79,77%	81,96%	82,83%	83,52%	84,09%
200	82,73%	81,84%	80,40%	82,41%	83,30%	83,74%
240	83,31%	82,61%	82,42%	80,73%	82,54%	83,48%
280	83,68%	83,14%	83,27%	82,60%	81,00%	82,77%
320	84,13%	83,80%	83,67%	83,53%	82,62%	81,29%

FONTE: O Autor (2014)

Assim, observou-se (tabela 18) que o recálculo dos custos atualizados pelo procedimento MudaArvoreOtimizada, fixando o número de origens ou destinos, gera economias de tempos maiores ainda para os problemas retangulares.

Os mesmos exemplos foram também resolvidos utilizando a lista *ReferenciaCAN* e o parâmetro *PercentualEconomiaAnterior* como sendo 0,79 e os percentuais de economia de tempo de resolução em relação ao A_Otimizado estão apresentados na tabela 19.

TABELA 19 – ECONOMIA DE TEMPO DE RESOLUÇÃO PELO MÉTODO LISTA_0,79 EM RELAÇÃO AO MÉTODO A_Otimizado

Origens \ Destinos	120	160	200	240	280	320
120	20,85%	30,53%	39,05%	48,10%	46,95%	50,72%
160	27,70%	25,90%	28,17%	39,97%	45,91%	49,94%
200	37,02%	27,44%	28,49%	29,87%	37,68%	45,23%
240	42,60%	37,79%	29,67%	32,90%	28,01%	35,53%
280	46,88%	45,20%	39,79%	30,19%	32,80%	32,85%
320	47,05%	48,66%	44,91%	36,69%	29,90%	33,12%

FONTE: O Autor (2014)

Na tabela 20 são apresentados os resultados de tempo de economia de resolução quando é comparada a utilização da lista *ReferenciaCAN* e do parâmetro *PercentualEconomiaAnterior*=0,79.

TABELA 20 – ECONOMIA DE TEMPO DE RESOLUÇÃO PELO MÉTODO LISTA_0,79 EM RELAÇÃO AO MÉTODO A_MODI

Origens \ Destinos	120	160	200	240	280	320
120	83,40%	87,47%	89,66%	91,36%	91,43%	92,04%
160	86,80%	85,01%	87,04%	89,69%	91,09%	92,04%
200	89,12%	86,82%	85,98%	87,66%	89,59%	91,09%
240	90,42%	89,18%	87,64%	87,07%	87,43%	89,35%
280	91,33%	90,76%	89,93%	87,85%	87,23%	88,43%
320	91,60%	91,68%	91,00%	89,57%	87,82%	87,49%

FONTE: O Autor (2014)

Poderia ser ainda realizada a comparação com a resolução pelo método MODI implementado em estrutura de quadro, mas optou-se por não fazê-la na

presente seção para não comparar ganhos de estrutura com os procedimentos de recálculo dos custos atualizados e de escolha da variável a entrar na base.

O desempenho da utilização da lista ReferenciaCAN e do parâmetro PercentualEconomiaAnterior está associado ao número de vezes que a lista é reiniciada (pois este processo exige que sejam analisados os valores de custos atualizados de cada variável não básica) e ao número médios de iterações (trocas de base) realizadas para cada lista. Para os mesmos exemplos desta seção, apresenta-se na tabela 21 o número médio de listas utilizadas para cada combinação de tamanho e na tabela 22 o número médio de iterações por lista.

TABELA 21 – NÚMERO MÉDIO DE LISTAS UTILIZADAS NO MÉTODO LISTA_0,79

Origens \ Destinos	120	160	200	240	280	320
120	71,84	59,89	60,89	56,79	60,47	64,47
160	66,74	96,74	86,11	81,79	76,47	76,47
200	63,74	89,74	113,58	110.63	103,32	97,00
240	67,26	79,11	102,16	142,16	128,79	119,84
280	62,79	83,05	87,63	120,63	164,42	147,47
320	65,89	77,84	99,58	126,16	151,11	188,11

FONTE: O Autor (2014)

TABELA 22 – NÚMERO MÉDIO DE ITERAÇÕES POR LISTA NO MÉTODO LISTA_0,79

Origens \ Destinos	120	160	200	240	280	320
120	4,54	7,78	9,25	10,49	10,57	10,45
160	6,96	4,96	7,62	9,42	11,31	11,84
200	9,01	7,41	5,48	7,71	9,83	11,76
240	9,55	9,66	8,04	5,61	8,02	10,30
280	10,40	10,66	10,05	8,32	5,74	8,32
320	10,44	11,69	11,05	9,89	7,98	5,98

FONTE: O Autor (2014)

Com os valores da tabela 22, tem-se que, fixado o número de origens ou destinos, o número médio de iterações por lista aumenta para problemas mais retangulares.

6 CONSIDERAÇÕES FINAIS E SUGESTÕES PARA TRABALHOS FUTUROS

Na presente tese foi apresentado um estudo relativo ao Problema de Transporte, um dos problemas clássicos da área de Pesquisa Operacional. A motivação inicial para o trabalho foi o Problema de Transporte com Custo Fixo e para isso foi estudado com maior profundidade o Problema de Transporte Clássico.

Diante dos resultados obtidos já para o Problema de Transporte Clássico, foi decidido delimitar a presente tese a este problema, sendo que o material desenvolvido para a abordagem do Problema de Transporte com Custo Fixo poderá ser utilizado para trabalhos futuros.

Para a resolução do PT, o método mais difundido é o MODI, sendo mais comum a abordagem da resolução dele em quadro. Entretanto, um PT pode ser também representado por um grafo bipartido e cada solução básica factível do PT em quadro é associada a uma árvore geradora.

Quando analisada a resolução em árvore geradora, o processo de troca de base é representado pela inclusão de um arco e exclusão de outro na árvore geradora. Uma vantagem da utilização da estrutura em árvore, especialmente para problemas grandes, é que o processo de encontrar o ciclo é mais rápido do que na abordagem em quadro.

Como a cada iteração é necessário encontrar o ciclo, existe possibilidade de ganho em termos de tempo computacional utilizando a abordagem em árvore e isto foi verificado nas implementações realizadas, onde observou-se uma redução média de 60,24% para Problemas de Transporte e 73,36% para Problemas de Designação (caso particular do Problema de Transporte em que as quantidades de oferta e demanda de cada nó são iguais a 1) na resolução em estrutura em árvore quando comparada a resolução em estrutura de quadro.

A parte com maior esforço computacional passou então a ser o cálculo dos novos valores de custos atualizados. Foi observada então a possibilidade de que não era necessário o recálculo de todos os custos atualizados. Além disso, para os que eram necessários serem recalculados, os novos valores poderiam ser obtidos por uma simples operação e sem a necessidade de cálculo dos valores das variáveis duais.

Utilizando na demonstração as variáveis duais é fácil provar que o cálculo delas não é necessário para a obtenção dos novos custos atualizados. Entretanto, seria incoerente o fato de usar as variáveis duais para demonstrar que elas não são necessárias.

Foi então demonstrado, utilizando somente os valores originais de custos, que, dados os custos atualizados numa determinada solução básica factível, a variável que entrará na base e a que sairá, é possível determinar cada um dos novos valores de custos atualizados com no máximo uma operação.

Com isso o procedimento MudaArvore foi alterado para o procedimento MudaArvoreOtimizada e esta consideração permitiu uma nova redução de tempo computacional, que em média foi de 80,78% para Problemas de Transporte e de 90,71% para Problemas de Designação em relação a implementação do método MODI em árvore. Quando comparada com a resolução em quadro pelo método MODI, a redução de tempo média foi de 92,34% para os Problemas de Transporte e de 97,51% para Problemas de Designação.

Ressalta-se que cada uma das implementações realizadas não altera o número de iterações necessárias para resolver cada problema e nem as variáveis que entram ou saem da base. Os ganhos computacionais são exclusivos da estrutura de armazenamento nos dados e procedimentos de atualização de valores de variáveis.

A estrutura de dados, assim como as rotinas computacionais, apresentadas na presente tese podem ainda facilmente ser adaptadas para o Problema de Transporte Esparso que em conjunto com os procedimentos descritos em Silva (2012) podem gerar bons resultados para o Problema de Transporte Esparso.

Além disso, a estrutura de dados, assim como as rotinas computacionais, apresentadas na presente tese podem facilmente ser adaptadas também para o Problema de Transporte com Custo Fixo.

Ainda sobre o PTCF, a demonstração realizada utilizando apenas os ciclos fornece ideias que podem ser utilizadas para demonstrar “quais quantidades” precisam ser recalculadas e como podem ser recalculadas de melhor forma.

Após a troca do procedimento MudaArvore por MudaArvoreOtimizada, o maior esforço computacional passou a ser a verificação de qual seria a variável a entrar na base. Diante disso, foi realizada a inclusão da variável *ReferenciaCAN*, resultando em novos ganhos computacionais, agora de, em média para os

problemas testados, de 22,48% para Problemas de Transporte e 47,78% para Problemas de Designação.

Por fim, foi incluído no algoritmo um parâmetro, denominado *PercentualEconomiaAnterior*, com o objetivo de reinicializar a lista *ReferenciaCAN* quando as economias geradas pelas variáveis referenciadas nela começassem a reduzir-se acima de um valor permitido. Por ser um parâmetro, foram realizados testes com valores no intervalo de possibilidades dele a fim de tentar identificar o comportamento do tempo de resolução para diferentes valores.

Com os resultados obtidos, observou-se que para todas as configurações de exemplos testados o melhor valor de *PercentualEconomiaAnterior* encontra-se no intervalo de 0,78 a 0,80. Ou seja, existe uma estabilidade do parâmetro e valores neste intervalo são os recomendados a serem utilizados para um melhor tempo de resolução.

Sugestões para trabalhos futuros

As estruturas de armazenamento de informações apresentadas no capítulo foram criadas considerando-se a possibilidade de adaptação para serem utilizadas para resolução do PTCF. Desta forma, aqui são apresentadas as adaptações que precisariam ser realizadas em cada uma das estruturas utilizadas para armazenamento de informações.

A estrutura **strTransporte** seria adaptada para a estrutura **strTransporteCustoFixo**, sendo que a única alteração seria a inclusão da variável *MatrizCustoFixo(,)*, conforme apresentado no quadro 28.

```
Estrutura strTransporteCustoFixo
    MatrizCustoVariavel(,)
    MatrizCustoFixo(,)
    CustoTotal
    VetorOfertas()
    VetorDemandas()
    SolucaoInicial(,)
    VetorU()
    VetorV()
Fim Estrutura
```

QUADRO 28 – ADAPTAÇÃO DA ESTRUTURA strTransporte PARA O PTCF

FONTE: O Autor (2014)

De forma similar a estrutura **strBasicas** seria adaptada para a estrutura **strBasicasCustoFixo**, apresentada no quadro 29, e passaria a ter também uma variável *CustoFixo*. Ressalta-se que está variável não seria rigorosamente necessária, uma vez que com as informações de ***strBasicas***.*Linha*, ***strBasicas***.*Coluna* e ***strTransporteCustoFixo***.*MatrizCustoFixo(,)* seria possível obter este valor, mas com a inclusão da variável *CustoFixo* na estrutura a **strBasicasCustoFixo** o acesso a informação fica mais rápido.

```
Estrutura strBasicasCustoFixo
    Linha
    Coluna
    Quantidade
    CustoFixo
    CustoVariavel
    CustoTotal
Fim Estrutura
```

QUADRO 29 – ADAPTAÇÃO DA ESTRUTURA strBasicas PARA O PTCF

FONTE: O Autor (2014)

O mesmo raciocínio explicado para a estrutura **strBasicasCustoFixo** vale também para a estrutura **strNaoBasicasCustoFixo**, apresentada no quadro 30. Além disso, como para o caso da existência do custo o fato de o custo atualizado da variável não básica ser negativo não garante a melhora na solução caso a variável torne-se básica, também faz para da estrutura para o caso do PTCF a variável *EconomiaTotal*, responsável por armazenar a diferença no valor da função objetivo para o caso da variável não básica tornar-se básica.

Para o PTCF, também apresenta vantagem computacional a utilização da variável *ReferenciaSaida*. Esta variável armazena a informação de qual posição s na variável ***NaoBasicas()*** está a variável que sai da base para a entrada da variável $x_{NaoBasicas(t).Linha,NaoBasicas(t).coluna}$ na base.

Ressalta-se aqui que a variável *ReferenciaSaida* não apresentava vantagem computacional (pelo contrário, piorava o desempenho computacional) quando utilizada no PT, pois o fato do custo atualizado ser negativo já era suficiente para saber se uma variável não básica garantiria melhora na função objetivo. Entretanto, para o PTCF, dada uma variável candidata a entrar na base, é necessário saber qual a variável que sai e qual a quantidade para que assim seja possível calcular o

valor de *EconomiaTotal* e possa ser avaliado se a troca das variáveis na base gera redução ou não no valor da função objetivo.

Também é interessante armazenar o valor *PontoEncontro* que representa a profundidade do ancestral comum de maior profundidade entre os nós $\mathcal{O}_{NaoBasicas(t).Linha}$ e $\mathcal{D}_{NaoBasicas(t).Coluna}$.

Por fim, gera vantagem computacional o armazenamento da informação *QuantidadeEntrada*, que representa o valor que a variável $x_{NaoBasicas(t).Linha,NaoBasicas(t).Coluna}$ entra na base.

A vantagem da utilização, para o caso do PTCF, das variáveis ***strNaoBasicasCustoFixo***.*ReferenciaSaida*, ***strNaoBasicas***.*PontoEncontro* e ***strNaoBasicasCustoFixo***.*QuantidadeEntrada* ocorre nos procedimentos para atualização somente das informações necessárias após cada pivoteamento. Da mesma forma que o recálculo apenas dos custos atualizados necessários gerou redução no tempo computacional para o PT, a atualização apenas das informações necessária também gera grande ganho de tempo no PTCF.

```
Estrutura strNaoBasicasCustoFixo
    Linha
    Coluna
    CustoVariavel
    CustoAtualizado
    CustoFixo
    QuantidadeEntrada
    EconomiaTotal
    ReferenciaSaida
Fim Estrutura
```

QUADRO 30 – ADAPTAÇÃO DA ESTRUTURA strNaoBasicas PARA O PTCF

FONTE: O Autor (2014)

Para o caso do PTCF também existe a garantia, sendo factível o problema, da existência de solução ótima num ponto extremo e que consequentemente poderá ser representada por uma árvore geradora.

No caso do PT, dimensionando as variáveis ***Linha()*** e ***Coluna()*** como sendo vetores do tipo **strNoArvore** era possível armazenar as informações referentes à arvore. Para o caso do PTCF a estrutura pode ser adaptada para

strNoArvoreCustoFixo, apresentada no quadro 32, adicionando a variável *ListaDeAncestrais*.

```
Estrutura strNoArvoreCustoFixo
    ListaDeFilhos
    Pai
    ListaDeAncestrais
Fim Estrutura
```

QUADRO 31 – ADAPTAÇÃO DA ESTRUTURA strNoArvore PARA O PTCF

FONTE: O Autor (2014)

No caso do PT, a lista de ancestrais era gerada duas vezes, uma para cada nó, a cada iteração. Para o PTCF seria necessário obter a lista de ancestrais para cada nó a cada iteração para o cálculo de ***NaoBasicas()***.*EconomiaTota*l. Por isso, armazenar a lista de ancestrais de cada nó e somente atualiza-las quando necessário também gera melhores tempos computacionais.

A ideia de explicar com detalhes a adaptação para PTCF das estruturas aqui utilizadas é facilitar o estudo de trabalhos futuros. Sobre os procedimentos de atualização das variáveis no PTCF foram encontradas possibilidades de economia de cálculos em alguns pontos e formadas conjecturas para outros, sendo um problema aberto para trabalhos futuros o desenvolvimento de procedimentos eficientes para a atualização das variáveis.

A existência de algoritmos rápidos para resolução tanto do PT como do PTCF, ou outras variações do Problema de Transporte, pode fazer com que sejam viáveis a resolução de problemas maiores que utilizam a resolução de Problemas de Transporte como subproblemas. Não é preocupação na presente elencar estes problemas, mesmo porque problemas que hoje não são modelados como PT podem vir a ser e caso isso aconteça, quem se interessar terá a disposição um procedimento eficiente de resolução.

Desta forma, as sugestões para trabalhos futuros constituem-se em duas partes: a primeira referente à adaptação de conceitos aqui utilizados para outras variações do Problema de Transporte, em especial para o PTCF, e a segunda para aplicações utilizando o Problema de Transporte como modelo base.

REFERÊNCIAS

ADLAKHA, V.; KOWALSKI, K. A simple heuristic for solving small fixed-charge transportation problems. **Omega**, v. 31, p. 205–211, 2003).

ADLAKHA, V.; KOWALSKI, K; LEV, B. A branching method for the fixed charge transportation problem. **Omega**, v. 38, p. 393–397, 2010.

AGARWAL, Y.; ANEJA, Y.. Fixed-charge transportation problem: Facets of the projection polyhedron. **Operations Research**, v. 60, n. 3, p. 638-654, 2012.

AGUADO, J. S. 2009. Fixed charge transportation problems: a new heuristic approach based on lagrangean relaxation and the solving of core problems. **Annals of Operations Research**, v. 172, n. 1, p. 45-69, 2009.

AMOIA, A.; COTTAFAVA, G. Invariance Properties of central trees, **IEEE Trans. Circuit Theory, CT-18**, 465-567, 1971.

ARAMUTHAKANNAN, S.; KANDASAMY, P. R. Revised Distribution Method of finding Optimal Solution for Transportation Problems. **IOSR Journal of Mathematics**, v. 4, n. 5, p. 39-42, jan-fev/2013.

ARENALES, M. N. *et al*. **Pesquisa Operacional**. Rio de Janeiro: Elsevier, 2007, p. 524.

BABU *et al*. A Simple Experimental Analysis on Transportation Problem: A New Approach to Allocate Zero Supply or Demand for All Transportation Algorithm. **Interantional Journal of Engineering Research and Applications**, v. 4, n. 1 (Version 2), p. 418-422, jan/2014.

BALINSKI, M. L. 1961. Fixed-cost transportation problems. **Naval Research Logistics Quarterly**, v. 8, n. 1, p. 41-54, 1961.

BARR, R.S., R.S. GLOVER F., KLINGMAN, D. A new optimization method for large scale fixed charge transportation problems. **Operations Research**, v. 29, n. 3, p. 448–463, mai-jun/1981.

BAZARAA, M. S.; JARVIS, J, J.; SHERALI, H. D. **Linear Programming and Network Flows**, 4 ed. John Wiley & Sons, 2010.

BEZRUKOV, S.; KADERALI, F.; POGUNTKE, W. On central spanning trees of a graph. **Lecture Notes in Computer Science**, Vol. 1120, 53-58, Springer, Berlin, 1996.

BOURGEOIS, F.; LASALLE, J. C. **An extension of the Munkres algorithm for the assignment problem to rectangular matrices**. Communications of the ACM 14, 1971.

BRITO, J. A. M., MONTENEGRO, F. M. T. e OCHI, L. S. Um algoritmo ILS para melhoria de eficiência da estratificação estatística. **Anais do XLI SBPO - Simpósio Brasileiro de Pesquisa Operacional**. 2009.

CABOT, A. V.; ERENGUC, S. S. Some branch and bound procedures for fixed cost transportation problems. **Naval Research Logistics Quarterly**, v. 31, n. 1, p. 145-154, 1984.

CARVALHO, L. E. X. Decomposição em Árvore de Grafos com Largura Limitada – Uma Pesquisa Algorítmica. 89 p. Dissertação – Ciência da Computação, UFC, 2002.

CHARNES, A.; COPPER, W. W. The stepping-stone method for explaining linear programming calculation in transportation problem. **Management Science**, v. 1, p. 49-69, 1954.

CHARTRAND, G.; OELLERMANN, O. R. **Applied and Algorithmic Graph Theory**. MCGraw-Hill, 1993.

CHEN, H. C.; WANG, Y. L. An Efficient Algorithm for Generating Prüfer Codes from Labelled Trees. **Theory of Computing Systems**, v. 33, p. 97-105, 2000.

CHRISTOFIDES, N. **Graph Theory: An Algorithmic Approach**. Academic Press: London, 1975

COOPER, L.; DREBES, C. An approximate solution method for the fixed charge problem. **Naval Research Logistics Quarterly**, v. 14, n. 1, 1967.

COOPER, L.; OLSEN, A .M. **Random perturbations and the MI-MII heuristics for the fixed-charge problem**. Report No. C00-1493-7. Department of Appllied Mathematics and Computer Science. Washington University.

CORMEN, T. H., LEISERSON, C. H., RIVEST R.L. **Algoritmos: Teoria e Prática**, 2. ed. Campus, 2002.

DENZLER, D. R. An approximate solution method for the fixed charge problem. **Naval Research Logistics Quarterly**, v. 6, n. 1, 1969.

DEO, N. A central tree. **IEEE Trans. Circuit Theory CT-13**. 439-440, 1966.

DIJKSTRA, E. W. A note on two problems in connexion with graphs. **Numerische Mathematik**, 1, p. 269–271, 1959.

DUTTON, R. *et al*. The Optimal Location of Nuclear Power Facilities in the Pacific Northwest. ***Operations Research***, 22, p. 478-487, 1974.

FLOYD, R. W. Algorithm 97: Shortest Path. **Communications of the ACM**, v. 5, n. 6, p. 345, jun/1962.

FORD L. R., FULKERSON D. R. Solving the transportation problem. **Management Science**, v 3, n. 1, out/1956.

FOX, B. L. Data Structures and Computer Science Techniques in Operations Research. **Operations Research**. v. 26 Issue 5, p. 686-717, 1978.

FRANK, R. S. **On the Fixed Charge Hitchcock Transportation Problem**. The Johns Hopkins University, 1972.

GIBBONS, A. **Algorithmic graph theory**. Cambridge University, 1985.

GLOVER, F.; AMINI, M.; KOCHENBERGER, G. Parametric ghost image processes for fixed-charge problems: A study of transportation networks. **Journal of Heuristics**, v. 11, n. 4, p. 307-336, 2005.

GOTHE-LUNDGREN, M.; LARSSON, T. A set covering reformulation of the pure fixed charge transportation problem. **Discr. Appl. Math**, v. 48, n. 3, p. 245-259, 1994.

GRAY, P. Exact solution of the fixed-charge transportation problem. **Operations Research**, v. 19, n. 6, p. 1529-1538, 1971.

HAKIM, M. A. An Alternative Method to Find Initial Basic Feasible Solution of a Transportation Problem. **Annals of Pure and Applied Mathematics**, n. 1, n. 2, p. 203-209, 2012.

HARRIS, J. M., HIRST, J. L., MOSSINGHOFF, M. J. **Combinatorics and Graph Theory**. Second Edition. Springer, 2008.

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

HITCHCOCK, F.L. (1941). The distribution of a product from several sources to númerous facilities. Journal of Mathematical Physics, n.20,p. 224-230.

HULTBERG, T. H.; CARDOSO, D. M. The teacher assignment problem: A special case of the fixed charge transportation problem, **European Journal of Operational Research**, v. 101, p. 463-473, 1997.

JI, P.; CHU, K. F. A dual-Matrix approach to the transportation problem. **Asia-Pacific Journal of Operation Research** 19(1) 35-45, 2002.

JO, J. B.; LI, Y.; GEN, M. Nonlinear fixed charge transportation problem by spanning tree-based genetic algorithm. **Computers & Industrial Engineering**, v. 53, n. 2, 290–298, 2007.

JOSHI, R. V. Optimization Techniques for Transportation Problems of Three Variables. **IOSR Journal of Mathematics**, v. 9, n. 1, p. 46-50, nov-dez/2013.

KADERALI, F. A counterexample to the algorithm of Amoia and Cottafava for finding central trees, **Technical Report FB 19, TH Darmstadt,** 1973.

KISHI, G. KAJITANI Y. Maximally distant trees and principal partition of a linear graph. **IEEE Trans. Circuit Theory. vCT-16**. 323-330, 1969.

KLOSE, A. Algorithms for solving the single-sink fixed-charge transportation problem. **Computers & Operations Research**, v. 35, p. 2079–2092, 2008.

KOOPMANS, T.C. Optimum utilization of the transportation system. **Econometrica**, v.17, p. 3-4. 1947.

KRUSKAL, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. **Proceedings of the American Mathematical Society**, 7, p. 48–50, 1956.

KUHN, H. W.; BAUMOL, W. J. An approximate algorithm for the fixed-charge transportation problem. **Naval Research Logistics Quarterly**, v. 9, n.1, p. 1-16, 1961.

LOTFI, M. M.; TAVAKKOLI-MOGHADDAM, R. A genetic algorithm using priority-based encoding with new operators for fixed charge transportation problems. **Appl. Soft Comp**, v. 13, n. 5, p. 2711-2726, 2013.

MALIK, N. On Deo's central tree concept. **IEEE Trans. Circuit Theory. vCT-15**. 283-284, 1968.

MURTY, K. G. Solving the fixed charge problem by ranking the extreme points. **Operations Research**, v. 16, n. 2, p. 268-279, 1968.

MURTY K. G. **Linear Programming**. New York: John Wiley e Sons, 1983.

O'CONNOR, D. R. **Algorithms and data structures**. Capítulo 4. ©2002. Disponível em. <http://www.derekroconnor.net/home/MMS406/Trees.pdf>. Acesso em: 19/07/2011.

PALANIYAPPA, R.; VINOBA, V. A NEW TYPE OF TRANSPORTATION PROBLEM USING OBJECT ORIENTED MODEL. **International Journal of Mathematical Archive**, v. 4, n. 11, p. 71-77, 2013.

PALEKAR U. S.; KARWAN, M. H.; ZIONTS, S. A Branch-and-Bound Method for the Fixed Charge Transportation Problem. **Management Science**, v. 36, n. 9, p. 1092-1105, set/1990.

PAPAMANTHOU, C., PAPARRIZOS, K., SAMARAS, N. Computational experience with exterior point algorithms for the transportation problem. **Applied Mathematics and Computation**, v. 158, p. 459–475, 2004.

PAPAMANTHOU, C., STEIGLITZ, K. **Combinatorial Optimization: Algorithms and Complexity**. Prentice-Hall, 1982.

PRIM, R. C. Shortest connection networks and some generalizations. **Bell System Technical Journal**, 36, p. 1389–1401, 1957.

PUCCINI A. L.; PIZZOLATO N. D. **Programação Linear**. Rio de Janeiro, 1987.

ROBERS, P.; COOPER, L. **A Study of The Fixed Charge Transportation Problem**. Department of Applied Mathematics and Computer Science, Washington University, 1969.

SACOMOTO, G. A. T. Árvores de Ukkonen: caracterização combinatória e aplicações, 75 p. Dissertação – IME, USP, São Paulo, 2011.

SADAGOPAN, S.; RAVINDRAN, A. A vertex ranking algorithm for the fixed-charge transportation problem, **Journal of Optimization Theory and Applications**, v. 37, p. 221-230, 1982

SHARMA, G.; ABBAS, S. H.; GUPTA, V. K. Solving Transportation Problem with the help of Integer Programming Problem. **IOSR Journal of Engineering**, v. 2, n. 6, p, 1274-1277, jun/2012.

SHARMA, J. K. Extensions and Special Cases of Transportation Problem: a Survey. **Digital Library of India**, p. 928-940, 1977.

SHINODA, S.; KAWAMOTO, T. On central trees of a graph. **Proceedings of the 17th Symposium of Research Institute of Electric Communication on Graph Theory and Algorithms**, p.137-151, 1980.

SHINODA, S.; SAISHU, K. Conditions for an incidence set to be a central tree, **Technical Report CAS80-6, Institute of Electrical and Communication Engineering, Japan**, 1980.

SILVA, T. C. L. **Nova Metodologia para Resolução de Problemas de Transporte em Casos Esparsos**. 122 p. Tese PPGMNE-UFPR, Curitiba, 2012.

SOUZA, D. O. **Algoritmos Genéticos Aplicados ao Planejamento do Transporte Principal de Madeira**. 169 p. Dissertação (Mestrado em Engenharia Florestal) – Universidade Federal do Paraná, Curitiba, 2004.

STEINBERG, D. I. The Fixed Charge Problem. **Naval Research Logistics Quarterly**, v. 17, n. 2, p. 217-236, 1970.

STROUP, J. W. Allocation of Launch Vehicles to Space Missions: A Fixed-Cost Transportation Problem. **Operations Research**, v. 15, n. 6, p. 1157-1163, 1967.

SUN, M., J. et al. A tabu search heuristic procedure for the fixed charge transportation problem. **European Journal of Operational Research**, v. 106 p. 441-456, 1998.

Sun, M., P. G. McKeown. Tabu search applied to the general fixed charge problem. **Annals of Operational Research**, v. 41, n. 14, p. 405-420, 1993.

SUSTARCIC, A. M. An Efficient Implementation of the Transportation Problem. Dissertação Masters in Mathematical Science, University of North Florida, 1999.

THOMPKINS, C. J. **Group Theoretic Structures in the Fixed Charge Transportation Problem**. Georgia Institute of Technology, 1971.

VALIENTE, G. **Algorithms on Trees and Graphs**. Springer, 2010.

WALKER, W. E. A Heuristic Adjacent Extreme Point Algorithm for the Fixed Charge Problem. **Management Science**, 22, p. 587-596, 1976.

WISTON W. L. Operations **Research – Applications and Algorithms**. 4. ed. Thomson Brooks, 2004.

WRIGHT, D., C. LANZENAUER, H. V. Solving the fixed charge problem with lagrangian relaxation and cost allocation heuristics. **European Journal of Operational Research**, 42, 304-312, 1989.

APÊNDICE

Assim como um problema de transporte clássico, um problema de transporte com custo fixo também pode ser representado por um problema em rede no qual os nós representam os pontos de origens e destinos e os arcos representam as ligações de cada origem para cada destino, sendo que o custo total de cada arco é dado pela soma do custo fixo f_{ij} associado a ele com a multiplicação do custo variável c_{ij} com a quantidade q_{ij} transportada.

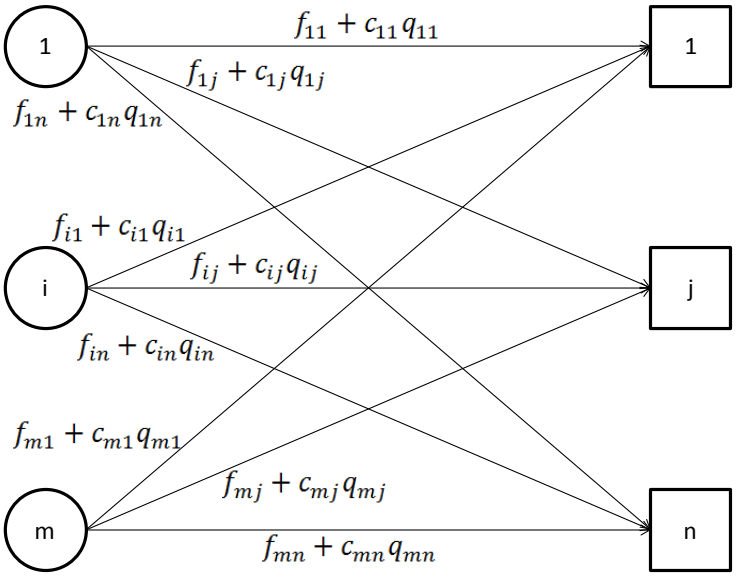


FIGURA 57 – REPRESENTAÇÃO EM REDE DO PROBLEMA DE TRANSPORTE COM CUSTO FIXO

FONTE: O Autor (2014)

Da mesma forma, assim como para o caso do problema de transporte clássico, todas as informações do Problema de Transporte com custo fixo podem ser representadas por meio de um quadro (quadro de transporte com custo fixo), no qual cada linha é relativa a uma origem e cada coluna a um destino. As informações de ofertas de cada origem e demandas de cada destino são apresentadas, respectivamente, na última coluna e na última linha. No presente trabalho é proposto que em cada célula estejam contidas as informações de custo fixo e custo unitário, conforme modelo exemplificado no quadro da figura 17.

	1		2		⋯	n		
1	f_{11}	c_{11}	f_{12}	c_{12}	⋯	f_{1n}	c_{1n}	a_1
2	f_{21}	c_{21}	f_{22}	c_{22}	⋯	f_{2n}	c_{2n}	a_2
⋮	⋮		⋮		⋮	⋮		⋮
m	f_{m1}	c_{m1}	f_{m2}	c_{m2}	⋯	f_{mn}	c_{mn}	a_m
	b_1		b_2		⋯	b_n		

FIGURA 58 – PROPOSTA DE REPRESENTAÇÃO EM QUADRO PARA UM PROBLEMA DE TRANSPORTE COM CUSTO FIXO

FONTE: O Autor (2014)

Como exemplo numérico para o quadro com informações relativas ao PTCF tem-se o quadro da figura 18. Este problema será utilizado como base para o exemplo descrito na presente seção.

	1		2		3		4		5		Ofertas
1	24	7	25	3	7	4	26	4	17	7	77
2	24	5	47	5	48	5	9	7	31	4	170
3	46	3	27	6	41	3	11	6	17	5	130
4	34	3	34	4	30	7	46	7	40	5	14
5	5	7	7	7	28	4	12	7	31	5	180
Demandas	74		133		13		186		165		571

FIGURA 59 – EXEMPLO PARA A PROPOSTA DE REPRESENTAÇÃO EM QUADRO PARA UM PROBLEMA DE TRANSPORTE COM CUSTO FIXO

FONTE: O Autor (2014)

Já para representar a solução atual a cada iteração, propões-se um quadro com uma estrutura para as células que representam variáveis não básicas de forma que nela estejam presentes as informações relevantes para o próximo pivoteamento. Para cada célula representante de uma variável não básica, propõe-se a consideração de seis informações: (r_s, t_s), $\bar{c}_{ij}$, θ_{ij}, A_{ij}, R_{ij} e Δ_{ij}, que representam, respectivamente, o valor que a variável x_{ij} assume caso torne-se básica, o arco que sai da base para a entrada do arco (i,j), o custo atualizado associado ao arco (i,j),

o aumento no custo fixo, a redução no custo fixo e a variação no valor da função objetivo caso a variável entre na base.

Cada célula associada a uma variável não básica é representado no quadro na forma

(r_s, t_s)	$\bar{c}_{ij}$	θ_{ij}
A_{ij}	R_{ij}	Δ_{ij}

Ressalta-se aqui que não necessariamente $A_{ij} = f_{ij}$ e $R_{ij} = f_{r_s t_s}$, uma vez que podem existir (ou surgir) variáveis básicas degeneradas e, por definição do PTCF, $x_{ij} = 0$ implica em $y_{ij} = 0$ e, consequentemente, $c_{ij}x_{ij} + f_{ij}y_{ij} = 0$. Desta forma, se na solução atual existe uma variável x_{bk} que é básica degenerada e ela assume um valor maior que zero após a entrada de x_{ij} na base, tem-se que $A_{ij} = f_{ij} + f_{bk}$. Agora se na solução atual existe uma variável x_{bk} que possui valor maior do que zero e ela torna-se básica degenerada, tem-se que $R_{ij} = f_{r_s t_s} + f_{bk}$.

Um exemplo do quadro que representa uma possível situação atual para o PTCF da figura 18 é apresentado na figura 19.

	1			2			3			4			5		
1	(3,1)	7	74	B			(3,3)	4	5	(3,4)	1	5	(1,2)	5	77
	24	46	**496**				7	41	**-14**	26	41	**-10**	17	25	**377**
2	(2,2)	3	5	B			(2,2)	3	5	(2,2)	2	5	B		
	24	47	**-8**				48	47	**16**	9	88	**-69**			
3	B			B			B			(3,3)	0	5	(2,3)	0	51
										11	41	**-30**	17	27	**-10**
4	(5,3)	-1	8	(5,3)	-3	8	(5,3)	3	8	B			(5,3)	-1	8
	34	28	**-2**	34	28	**-18**	30	28	**-26**				40	28	**4**
5	(5,3)	3	8	(5,2)	0	8	B			B			(5,3)	-1	8
	5	28	**1**	7	28	**-21**							31	28	**-5**

FIGURA 60 – EXEMPLO DE QUADRO PARA UMA SOLUÇÃO DO PROBLEMA DE TRANSPORTE COM CUSTO FIXO

FONTE: O Autor (2014)

O valor calculado para Δ_{ij} representa uma generalização do valor $\theta_{ij}\bar{c}_{ij}$ que era calculado para o caso do PT. Desta forma um algoritmo intuitivo para o PTCF seria adaptar o método MODI substituindo a escolha da variável não básica que possui o valor de $\bar{c}_{ij}$ mais negativo pela escolha da variável não básica que possui o valor de Δ_{ij} mais negativo. O procedimento de calcular Δ_{ij} para as variáveis não

básicas e utiliza-los como critério para decisão da variável a entrar na base já foi utilizado anteriormente por Sun *et al*. (1998).

Como uma solução básica é um ponto extremo, quando não existisse mais, após um número finito de iterações, valor de Δ_{ij} negativo, a solução seria um minimizador na vizinhança da base atual. Porém, enquanto para o PT o fato de não existir $\bar{c}_{ij}$ negativo era condição suficiente de otimalidade, para o caso do PTCF o fato de não existir Δ_{ij} negativo não é condição suficiente de otimalidade.

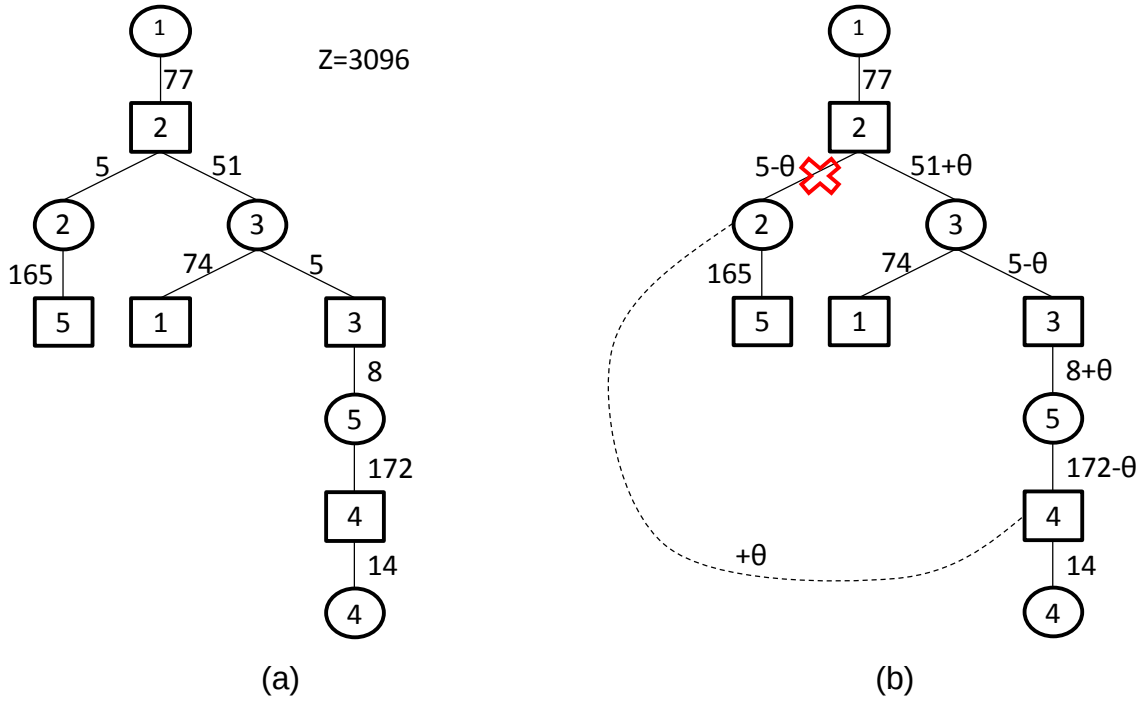


FIGURA 61 – EXEMPLO DE ÁRVORE PARA UMA SOLUÇÃO DO PROBLEMA DE TRANSPORTE COM CUSTO FIXO

FONTE: O Autor (2014)

A solução básica factível apresentada em quadro na figura 19 pode ser representada pela árvore da figura 20(a). Para a variável x_{24} tornar-se básica, a variável x_{22} deixa de ser básica[25] e x_{24} passa a valer 5 (figura 20(b)). A única variável que deixa de ser zero para assumir algum valor positivo é a x_{24}, assim $A_{24} = f_{24} = 9$. Por outro lado, as variáveis x_{22} e x_{33}, que possuíam valores positivos, passam a assumir o valor de zero, assim $R_{24} = f_{22} + f_{33} = 88$. Já a variação no custo variável é de $\theta_{24}\bar{c}_{24} = 5(2) = 10$. Desta forma, $\Delta_{24}= 10 + 9 - 88 = -69$. Após

[25] No caso de mais de uma possibilidade de variável para sair da base, sempre foi escolhida a primeira no caminho de O_i até D_j.

a variável x_{24} tornar-se básica no lugar da variável x_{22}, tem-se o quadro e a árvore referentes à solução básica factível apresentados nas figuras 21 e 22(a), respectivamente.

	1			2			3			4			5		
1	(3,1)	7	74	B			(3,3)	4	0	(3,3)	1	0	(3,3)	7	0
	24	46	**496**				7	0	**7**	26	0	**26**	17	0	**17**
2	(2,4)	1	5	(2,4)	-2	5	(2,4)	1	5	B			B		
	61	9	**57**	88	9	**69**	48	9	**44**						
3	B			B			B			(3,3)	0	0	(3,3)	2	0
										11	0	**11**	17	0	**17**
4	(5,3)	-1	13	(5,3)	-3	13	(5,3)	3	13	B			(4,4)	1	14
	75	28	**34**	75	28	**8**	30	28	**41**				40	46	**8**
5	(5,3)	3	13	(5,3)	0	13	B			B			(2,5)	1	165
	46	28	**57**	48	28	**20**							31	31	**165**

FIGURA 62 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A PRIMEIRA ITERAÇÃO

FONTE: O Autor (2014)

Como não existe Δ_{ij} negativo para a solução atual, tem-se um mínimo local. Entretanto, como já mencionado, este fato não é suficiente para garantir que esta solução seja a ótima para o problema. Neste sentido, uma possibilidade para buscar uma solução melhor é forçar uma variável não básica a entrar na base, mesmo isso gerando uma piora momentânea no valor da função objetivo, para a partir dela realizar novamente o processo de busca local. No presente exemplo será forçada a variável x_{41} a tornar-se básica.

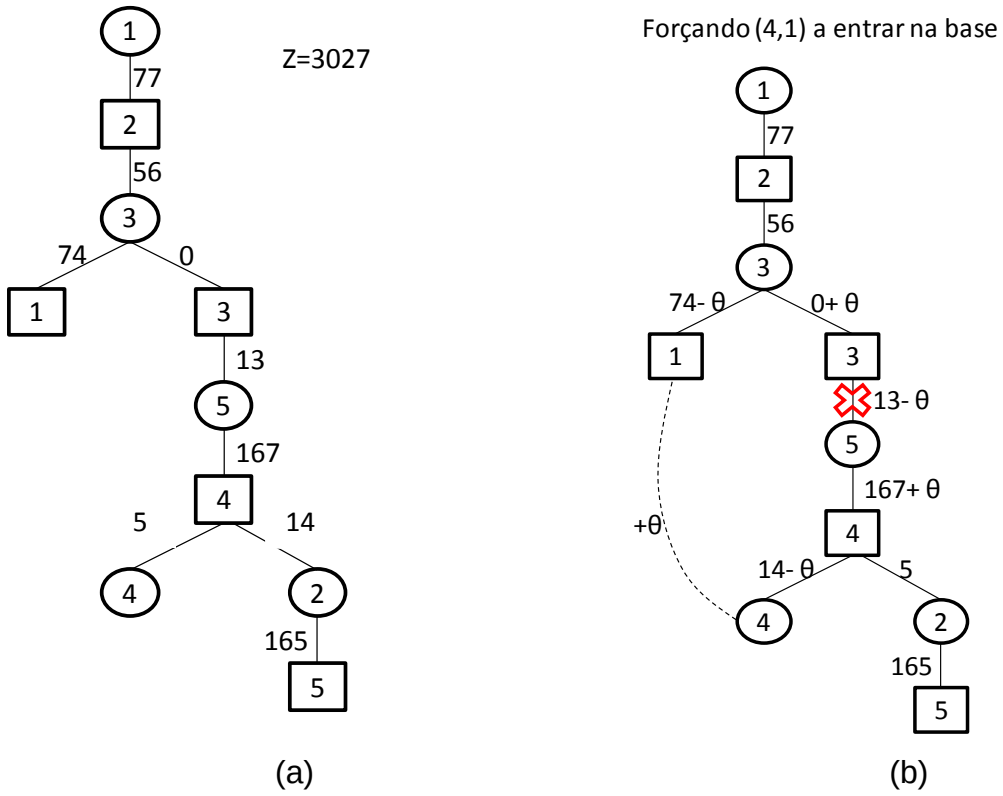


FIGURA 63 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A PRIMEIRA ITERAÇÃO

FONTE: O Autor (2014)

Para a variável x_{41} tornar-se básica, tem-se que $\bar{c}_{41} = -1$, $\theta_{41} = 13$, $A_{41} = f_{41} + f_{33} = 75$ e $R_{41} = f_{53} = 28$. Logo $\Delta_{41} = 13(-1) + 75 - 28 = 34$. A nova solução básica factível é representada nas figuras 23 e 24(a).

	1			2			3			4			5		
1	(3,1)	7	61	B			(3,3)	4	13	(4,4)	0	1	(4,4)	6	1
	24	46	**405**				7	41	**18**	26	46	**-20**	17	46	**-23**
2	(2,4)	2	5	(2,4)	-1	5	(2,4)	2	5	B			B		
	24	9	**25**	47	9	**33**	48	9	**49**						
3	B			B			B			(4,4)	-1	1	(4,4)	1	1
										11	46	**-36**	17	46	**-28**
4	B			(4,1)	-2	13	(3,3)	4	13	B			(4,4)	1	1
				34	34	**-26**	30	34	**48**				40	46	**-5**
5	(4,1)	4	13	(4,1)	1	13	(3,3)	1	13	B			(2,5)	1	165
	5	34	**23**	7	34	**-14**	28	75	**-34**				31	31	**165**

FIGURA 64 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A SEGUNDA ITERAÇÃO

FONTE: O Autor (2014)

Após a variável x_{41} tornar-se básica, houve um aumento de 34 no valor da função objetivo. Porém, para a nova solução existem possibilidades de melhorias. Por coincidência, para este exemplo, existe um valor de Δ (Δ_{34}) que já compensa o aumento na função objetivo gerado pela iteração anterior, mas isso não é regra que sempre ocorrerá.

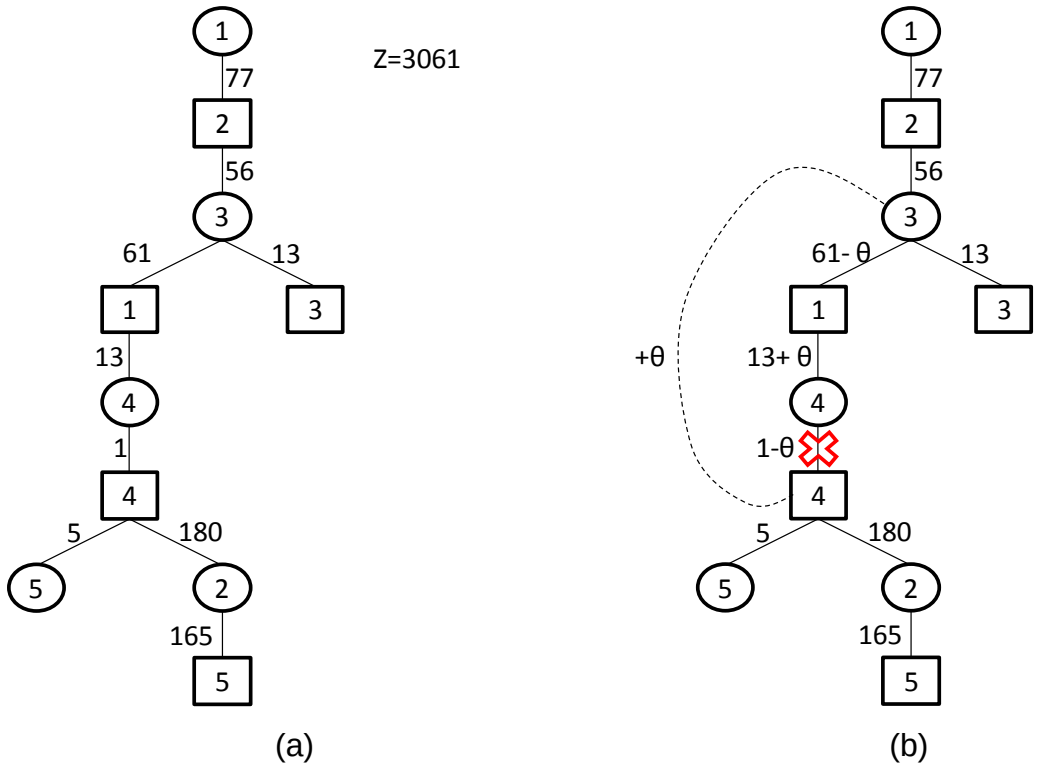


FIGURA 65 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A SEGUNDA ITERAÇÃO

FONTE: O Autor (2014)

Agora, para a entrada da variável x_{34} na base, tem-se que $\bar{c}_{34} = -1$, $\theta_{34} = 1$, $A_{34} = f_{34} = 11$ e $R_{34} = f_{44} = 46$. Logo $\Delta_{41} = 1(-1) + 11 - 46 = -36$. A nova solução básica factível é representada nas figuras 25 e 26(a).

	1			2			3			4			5		
1	(3,1)	7	60	B			(3,3)	4	13	(4,1)	1	1	(3,4)	7	1
	24	46	**398**				7	41	**18**	26	11	**16**	17	11	**13**
2	(2,4)	1	5	(2,4)	-2	5	(2,4)	1	5	B			B		
	24	9	**20**	47	9	**28**	48	9	**44**						
3	B			B			B			B			(3,4)	2	1
													17	11	**8**
4	B			(4,1)	-2	14	(3,3)	4	13	(3,4)	1	1	(3,4)	2	1
				34	34	**-28**	30	41	**41**	46	11	**36**	40	11	**31**
5	(3,1)	3	60	(3,2)	0	56	(3,3)	0	13	B			(2,5)	1	165
	5	46	**139**	7	27	**-20**	28	41	**-13**				31	31	**165**

FIGURA 66 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A TERCEIRA ITERAÇÃO

FONTE: O Autor (2014)

Após a variável x_{34} tornar-se básica, houve um decréscimo de 36 no valor da função objetivo e continuaram existindo valores de Δ negativos (figura 66). Seguindo o mesmo raciocínio, a próxima variável que entrará na base será a x_{42}.

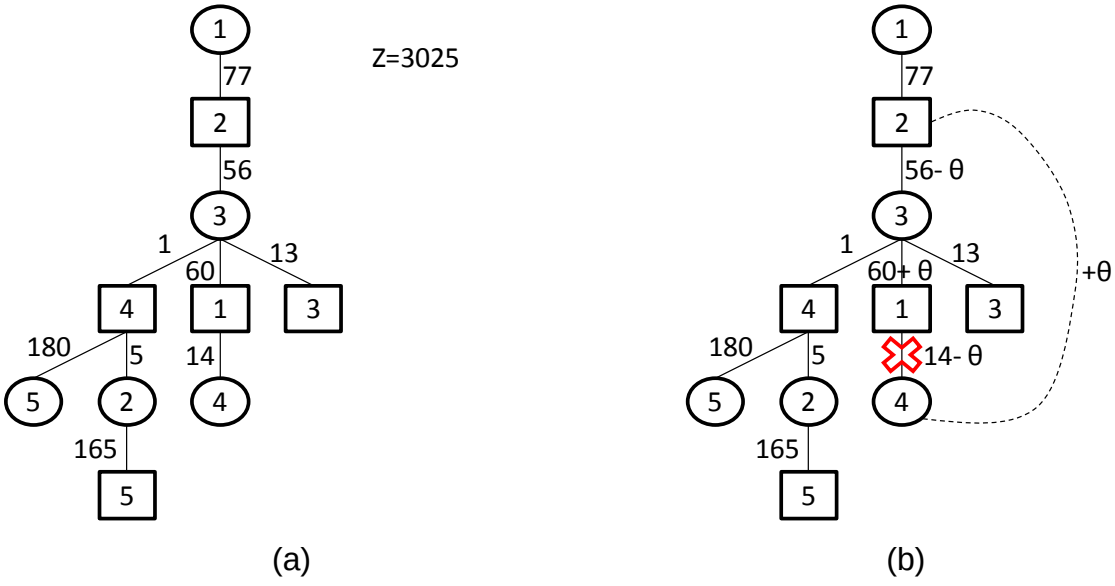


FIGURA 67 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A TERCEIRA ITERAÇÃO

FONTE: O Autor (2014)

Na solução atual, para a entrada da variável x_{42} na base, tem-se que $\bar{c}_{42} = -2$, $\theta_{42} = 14$, $A_{42} = f_{42} = 34$ e $R_{42} = f_{41} = 34$. Logo $\Delta_{41} = 14(-2) + 34 - 34 = -28$. A nova solução básica factível é representada nas figuras 27 e 28(a).

	1			2			3			4			5		
1	(3,1)	7	74	B			(3,3)	4	13	(3,4)	1	1	(3,4)	7	1
	24	46	**496**				7	41	**18**	26	11	**16**	17	11	**13**
2	(2,4)	1	5	(2,4)	-2	5	(2,4)	1	5	B			B		
	24	9	**20**	47	9	**28**	48	9	**44**						
3	B			B			B			B			(3,4)	2	1
													17	11	**8**
4	(4,2)	2	14	B			(3,3)	6	13	(3,4)	3	1	(3,4)	4	1
	34	34	**28**				30	41	**67**	46	11	**38**	40	11	**31**
5	(3,1)	3	74	(3,2)	0	42	(3,3)	0	13	B			(2,5)	1	165
	5	46	**181**	7	27	**-20**	28	41	**-13**				31	31	**165**

FIGURA 68 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A QUARTA ITERAÇÃO

FONTE: O Autor (2014)

Após a variável x_{42} tornar-se básica, houve um decréscimo de 28 no valor da função objetivo e continuaram existindo valores de Δ negativos (figura 68). Seguindo o mesmo raciocínio, a próxima variável que entrará na base será a x_{52}.

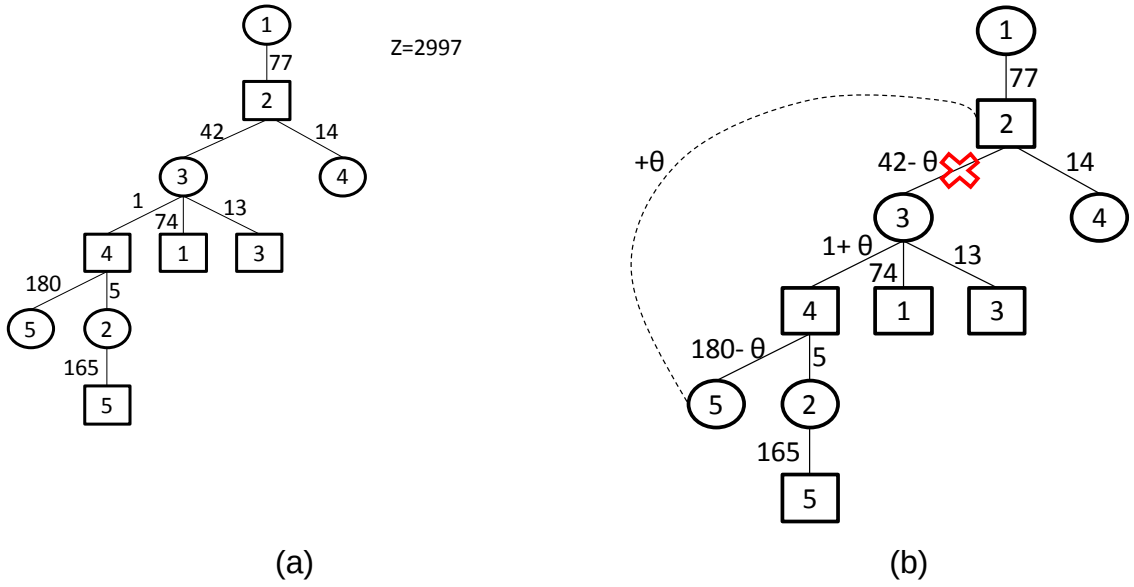


FIGURA 69 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A QUARTA ITERAÇÃO

FONTE: O Autor (2014)

Na solução atual, para a entrada da variável x_{52} na base, tem-se que $\bar{c}_{52} = 0$, $\theta_{52} = 42$, $A_{52} = f_{52} = 7$ e $R_{52} = f_{32} = 27$. Logo $\Delta_{52} = 42(0) + 7 - 27 = -20$. Neste caso, a redução no valor da função objetivo ocorre exclusivamente pela redução do custo fixo. A nova solução básica factível é representada nas figuras 29 e 30(a).

	1			2			3			4			5		
1	(3,1)	7	74	B			(3,3)	4	13	(1,2)	1	77	(1,2)	7	77
	24	46	**-496**				7	41	**-18**	26	25	**-78**	17	25	**-531**
2	(2,4)	1	5	(2,4)	-2	5	(2,4)	1	5	B			B		
	24	9	**-20**	47	9	**-28**	48	9	**-44**						
3	B			(5,2)	0	42	B			B			(3,4)	2	43
				27	7	**-20**							17	11	**-92**
4	(4,2)	2	14	B			(3,3)	6	13	(4,2)	3	14	(4,2)	4	14
	34	34	**-28**				30	41	**-67**	46	34	**-54**	40	34	**-62**
5	(3,1)	3	74	B			(3,3)	0	13	B			(5,4)	1	138
	5	46	**-181**				28	41	**13**				31	12	**-157**

FIGURA 70 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A QUINTA ITERAÇÃO

FONTE: O Autor (2014)

Após a variável x_{52} tornar-se básica, houve um decréscimo de 20 no valor da função objetivo e continuaram existindo valores de Δ negativos (figura 70). Seguindo o mesmo raciocínio, a próxima variável que entrará na base será a x_{53}.

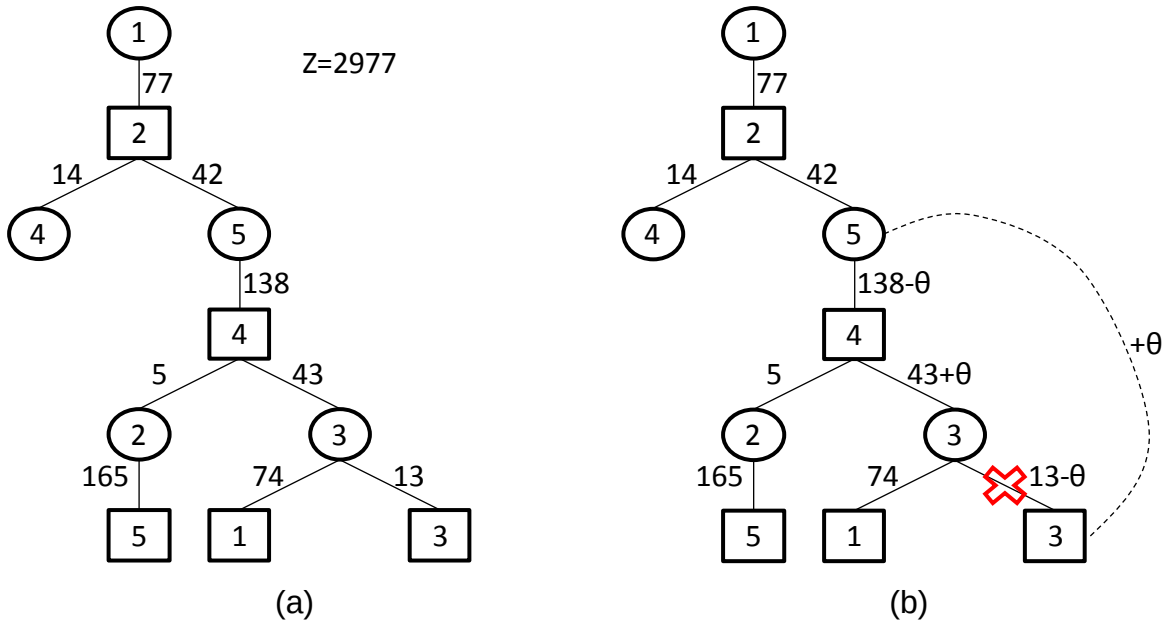


FIGURA 71 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A QUINTA ITERAÇÃO

FONTE: O Autor (2014)

Na solução atual, para a entrada da variável x_{53} na base, tem-se que $\bar{c}_{53} = 0$, $\theta_{53} = 13$, $A_{53} = f_{53} = 28$ e $R_{53} = f_{33} = 41$. Logo $\Delta_{53} = 13(0) + 28 - 41 = -13$. Novamente, a redução no valor da função objetivo ocorre exclusivamente pela redução do custo fixo. A nova solução básica factível é representada nas figuras 31 e 32.

	1			2			3			4			5		
1	(3,1)	7	74	B			(3,3)	4	13	(1,2)	1	77	(1,2)	7	77
	24	46	**-496**				7	41	**-18**	26	25	**-78**	17	25	**-531**
2	(2,4)	1	5	(2,4)	-2	5	(2,4)	1	5	B			B		
	24	9	**-20**	47	9	**-28**	48	9	**-44**						
3	B			(5,2)	0	42	(3,3)	0	13	B			(3,4)	2	43
				27	7	**-20**	41	28	**-13**				17	11	**-92**
4	(4,2)	2	14	B			(3,3)	6	13	(4,2)	3	14	(4,2)	4	14
	34	34	**-28**				30	41	**-67**	46	34	**-54**	40	34	**-62**
5	(3,1)	3	74	B			B			B			(5,4)	1	138
	5	46	**-181**										31	12	**-157**

FIGURA 72 – QUADRO REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A SEXTA ITERAÇÃO

FONTE: O Autor (2014)

Agora novamente tem-se um mínimo local, mas com valor de função objetivo menor do que o último mínimo local avaliado. Da mesma forma que na situação anterior, não existem garantias de que esta nova solução seja o ótimo global para o problema.

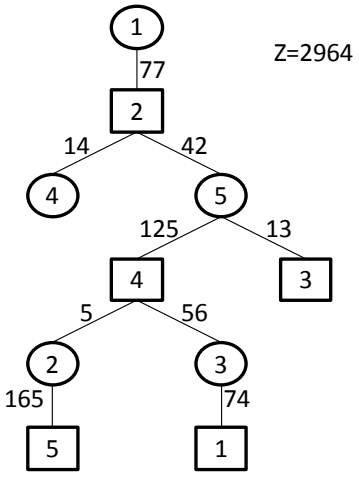


FIGURA 73 – ÁRVORE REFERENTE À SOLUÇÃO PARA O EXEMPLO DO PTCF APÓS A SEXTA ITERAÇÃO

FONTE: O Autor (2014)

O processo de realizar uma perturbação numa solução s^* que era um mínimo local, obtento a solução s' e, a partir da nova solução s' realizar novamente o processo de busca local convergiu para uma solução $s^{*\prime}$ melhor que a solução s^*. Poderia também acontecer de $s^{*\prime}$ ser pior que s^*, mas é possível concluir que a perturbação permite a possibilidade de soluções melhores virem a ser encontradas. Para o exemplo, o valor de $z = 2964$ coincide com a solução ótima para o problema.

Em diversas meta-heurísticas é trabalhada a ideia de um processo de busca local associado a um processo de perturbação em soluções já encontradas. Em particular, uma metodologia, segundo Brito, Montenegro e Ochi (2009), que vem sido utilizada em diferentes problemas e obtendo bons resultados é o *Iterated Local Search* (ILS), sendo o algoritmo básico desta meta-heurística apresentado no quadro 1.

Solução $s_0 = GerarSolucaoInicial(\quad)$
Solução $s^* = BuscaLocal(s_0)$
Enquanto $critério\ de\ parada\ não\ for\ atingido$ faça
Solução $s' = Perturbação(s^*, HistóricoDeBusca)$
Solução $s^{*\prime} = BuscaLocal(s')$
$s^* = CritérioDeAceitação(s^*, s^{*\prime}, HistóricoDeBusca)$
Fim enquanto
Retorne s^*

QUADRO 32 – DESCRIÇÃO DA META-HEURÍSTICA ILS
FONTE: Adaptado de Brito, Montenegro e Ochi (2009)

Sun *et al*. (1998) utilizaram a meta-heurística Busca Tabu e nela os valores de Δ_{ij} para variáveis não básicas são calculados diversas vezes ao longo do algoritmo proposto.

Métodos propostos que utilizaram a adaptação do MODI como algoritmo de busca local e procedimentos para mudanças de vértices quando um mínimo local era encontrado, como o proposto em Sun *et al*. (1998), apresentaram valores de função objetivo perto do ótimo. Desta forma, existe o desafio de fazer com que a busca local seja mais rápida a fim de reduzir o tempo de resolução.

Na presente tese, o objetivo não é propor uma nova metodologia de $CritérioDeAceitação$, $Perturbação$ ou $HistóricoDeBusca$, mas sim propor melhorias no processo de $BuscaLocal$ para que soluções possam ser encontradas em menor tempo computacional e também para tornar viáveis novos $CritérioDeAceitação$, novos processos de $Perturbação$ e novos critérios de parada, uma vez que o maior esforço computacional ocorre no processo de $BuscaLocal$. No próprio trabalho de Sun *et al*. (1998), após a mudança de base ocorre recálculo apenas parte dos valores de Δ's e os autores comentam que isto é realizado para menor tempo computacional.

6.1 Trabalhos sobre resolução para o Problema de Transporte com Custo Fixo

Uma primeira tentativa de método para obtenção de solução exata para o PTCF foi proposta por Murty (1967), sendo um algoritmo baseado em *ranking* de pontos extremos. Entretanto o algoritmo pode não ser eficiente para o caso de degenerescência na solução ótima e amplitude de custos variáveis maior que amplitude de custos fixos. Além disso, o método não é viável para problemas de dimensões maiores.

Uma segunda tentativa de método exato foi proposta por Gray (1968). O método consistia em decompor o problema num problema de programação inteira e numa série de subproblemas de transporte.

Ahrens e Finke (1975) apresentaram um método de perturbação para transformar um PTCF F num PTCF F' sem a ocorrência de variáveis básicas degeneradas. A resolução é então realizada em F' e após conhecida a solução de F', a solução de F pode ser resolvida por simples arredondamento.

Barr, Glover e Kligman (1980) propuseram um algoritmo de *branch-and-bound* para a resolução de PTCF esparsos. Testes foram realizados em problemas de $100x300$ com densidade de 10%, ou seja, 3000 arcos.

Sadagopan e Ravindran (1982) apresentaram um algoritmo baseado em *ranking* de pontos extremos, baseados no método de pontos extremos de Murty

(1968). A mudança constituiu em implementar atualizações dinâmicas e assim foi possível obter melhores valores de limite inferior.

Palekar, Karwan e Zionts (1990) desenvolveram novas condições de penalidade para o PTCF, mais fortes do que as de Driebeek[26] (1966) e Cabot (1984). Foram realizados testes em problemas de até 1200 arcos e observou-se redução no tempo computacional para resolução.

Göthe-Lundgreen e Larsson (1994) reformularam o PTCF como um problema de cobertura de conjuntos, propondo dois procedimentos de geração de restrições, sendo que um deles gera valores de limite inferior e superior para a solução e outro gera valores de limite inferior.

Hultberg e Cardoso (1997) estudaram o problema de alocação de professores como um caso especial do PTCF. Para a resolução, desenvolveram um procedimento de *branch-and-bound* que apresentou resultados melhores do que o cplex. Segundo os autores, era esperado, por ser um problema de otimização combinatória, que um algoritmo especializado apresentasse melhores resultados.

Uma primeira abordagem para problemas gerais de custo fixo utilizando a Busca Tabu foi realizada por Sun e McKeown (1993). Alguns anos mais tarde, Sun *et al*. (1998) desenvolveram uma abordagem utilizando Busca Tabu para a solução do PTCF, utilizando memória recente e memória baseada em frequência, junto com estratégias para memória de médio e longo prazo. Além disso, foi utilizado um procedimento de busca local que é uma adaptação do método MODI. Este procedimento é baseado no fato de existir uma solução ótima para o PTCF num ponto extremo do conjunto de restrições.

No caso do PTCF, para que ocorra uma redução no valor da função objetivo, não é possível utilizar a simples ideia de $\bar{c}_{ij} < 0$, uma vez que deve ser observada a variação no custo variável e também no custo fixo. No caso de variáveis que possuíam valor zero e passam a ter valores positivos, tem-se aumento no valor do custo fixo. Por outro lado, no caso de variáveis que possuíam valores positivos e passam a assumir o valor zero, ocorre diminuição no custo fixo. Além disso, existe uma variação no custo variável de $\delta\bar{c}_{ij}$, onde δ é o valor que a variável não básica x_{ij} assume ao entrar na base.

[26] O trabalho de Driebeek (1966) aborda problemas gerais de Programação Inteira Mista

$$\Delta_{ij}= \begin{cases} 0, se\ \delta = 0 \\ \delta\left(c_{ij} - \pi_i - \pi_j\right) + f_{ij} - \sum_{(p,q)\in U_1} f_{pq} + \sum_{(p',q')\in U_2} f_{p'q'}, se\ \delta > 0 \end{cases}$$

Onde $\pi_i's$ são as variáveis duais[27], $U_1 = \{(p,q)|(q,p) \in P\ e\ x_{pq} = \delta\}$, $U_2 = \{(p',q')|(p',q') \in P\ e\ x_{p'q'} = 0\}$ e P é o caminho de i até j em $\mathcal{T}$.

Ou seja, $\delta\left(c_{ij} - \pi_i - \pi_j\right)$ representa a variação referente aos custos variáveis, f_{ij} o custo referente a variável x_{ij} tornar-se maior do que zero e, consequentemente, y_{ij} precisar assumir o valor 1, $\sum_{(p,q)\in U_1} f_{pq}$ a variação no custo fixo das variáveis básicas que tornaram-se degeneradas e, por fim, $\sum_{(p',q')\in U_2} f_{p'q'}$ a variação no custo fixo causada pelas variáveis básicas que deixaram de ser degeneradas.

Dentro do algoritmo de Busca Tabu proposto, o método de busca local consiste em encontrar $(i,j) \notin \mathcal{T}$ tal que $\Delta_{ij}= min\{\Delta_{pq}|(p,q) \notin \mathcal{T}\}$. Ou seja, encontrar a variável não básica que gera a maior redução no valor da função objetivo quando torna-se básica.

Para testar o método foram gerados exemplos de diferentes tamanhos, de $10x10$ a $50x100$, com custos variáveis no intervalo de 3 a 8 e diferentes amplitudes de custos fixos. Os resultados obtidos foram comparados aos do procedimento de Steinberg (1977) e os resultados foram melhores em valores de função objetivo e tempo computacional.

Adlakha e Kowalski (2003) apresentaram uma heurística simples para a resolução de PTCF pequenos, sendo apresentados dois exemplos didáticos do método, um $4x3$ e outro $3x4$. O algoritmo proposto é baseado na relaxação de Balinski e no método de Vogel.

Kowalski (2004) apresentou propriedades que devem ser consideradas para métodos atuais e novos para a resolução do PTCF, especialmente sobre degenerescência. O autor cita que os únicos métodos existentes para ótimo global do problema são inviáveis computacionalmente para dimensões grandes.

Glover, Amini, Kochenberger (2005) apresentaram a heurística *Parametric Ghost Image Processes for Fixed-Charge Problems* (GIP) utilizando o FCTP como problema para estudo e os mesmos exemplos de Sun *et al*. (1998) para análise do

[27] No trabalho de Sun *et al*. (1998) os nós de origens são numerados de 1 até m e os nós de destino de m+1 até m+n. Desta forma, se $i \leq m$ então $\pi_i = u_i$ e se $i > m$ então $\pi_i = v_{i-m}$.

método proposto. Depois de realizados os testes computacionais, observou-se que o GIP obteve resultados melhores que o da Busca Tabu para quase todos os exemplos testados.

Jo, Li e Gen (2007) utilizaram algoritmo genético para resolução do PTCF, utilizando o número de Prüfer como cromossomo. O maior problema da utilização da codificação de Prüfer é que nem toda árvore é possível de ser associada a uma solução factível. Neste sentido, os autores propõem um procedimento de reparo no cromossomo para corrigir a infactibilidade. Entretanto, não foram apresentados resultados computacionais para o método proposto, apenas os resultados[28] para um problema $4x5$ e o outro $5x10$.

Hajiaghaei-Keshteli, Molla-Alizadeh-Zavardehi e Tavakkoli-Moghaddam (2010) utilizaram algoritmo genético com base na representação da solução básica em árvore e na representação de árvore pelo número de Prüfer. Nos 28 problemas testados, as soluções ficaram entre $3{,}75\%$ e $3{,}95\%$ do ótimo.

Adlakha e Kowalski (2010) desenvolveram uma heurística para o PTCF, utilizando como base o método de solução aproximada de Balinski. Os autores citam que o método heurístico é de fácil implementação para problemas de pequenos a médios, mas não apresentam resultados computacionais.

Adlakha, Kowalski e Lev (2010) desenvolveram um método exato baseado em ramificação e no cálculo de limites inferiores e superiores. Entretanto, não são apresentados resultados computacionais para problemas maiores, apenas um exemplo acadêmico de $4x3$.

Agarwal e Aneja (2012) estudaram a estrutura de projeção do poliedro do PTCF no espaço das variáveis binárias associadas aos custos fixos. Utilizando os resultados obtidos, foram desenvolvidos procedimentos heurísticos para a resolução do PTCF.

Lotfi e Tavakkoli-Moghaddam (2012) apresentaram uma adaptação para a estrutura do PTCF da codificação apresentada por Gin e Lin (2006) para problemas de transporte envolvendo dois estágios. Os autores utilizaram o número de Prüfer para representar o cromossomo no algoritmo genético e também desenvolveram novos operadores. Testes computacionais foram realizados para problemas de tamanhos $4x5$ a $30x50$.

[28] Em Coments on *"Nonlinear fixed charge transportation problem by spanning tree-based genetic algorithm"* é apresentada uma errata sobre os resultados.

Adlakha *et al*. (2013) apresentaram uma nova aproximação para o cálculo de limites inferiores para o PTCF. Testes computacionais foram realizados para problemas de tamanho $6x6$, $10x10$ e $10x15$, sendo os melhores resultados obtidos para os problemas menores. Porém, mesmo quando a solução ficou mais distante do ótimo, ela pode ser utilizada como uma boa solução inicial para outros métodos já presentes na literatura.