PROGRAMAÇÃO DE TABELAS PARA TORNEIOS ROUND ROBIN SIMPLES COM ESTÁDIOS PREDEFINIDOS

FABRÍCIO NUNES DA COSTA

ORIENTADOR: SEBASTIÁN ALBERTO URRUTIA

PROGRAMAÇÃO DE TABELAS PARA TORNEIOS ROUND ROBIN SIMPLES COM ESTÁDIOS PREDEFINIDOS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Belo Horizonte

Maio de 2009

Costa, Fabrício Nunes da
C837p Programação de Tabelas para Torneios Round Robin Simples com Estádios Predefinidos / Fabrício Nunes da Costa. — Belo Horizonte, 2009
xvi, 56 f. : il. ; 29cm

Dissertação (mestrado) — Universidade Federal de Minas Gerais
Orientador: Sebastián Alberto Urrutia

1. Programação de Tabelas. Teses. 2. Pesquisa Operacional - Teses. I. Título.

CDU 519.6*64

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Programação de Tabelas para Torneios Round Robin Simples com Estádios Predefinidos

FABRÍCIO NUNES DA COSTA

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. SEBASTIÁN ALBERTO URRUTIA - Orientador
Departamento de Ciência da Computação - UFMG

PROF. CELSO DA CRUZ CARNEIRO RIBEIRO
Departamento de Computação - UFF

PROF. MARCONE JAMILSON FREITAS SOUZA
Departamento de Computação - ICEB - UFOP

PROF. ALEXANDRE SALLES DA CUNHA
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 15 de maio de 2009.

Agradecimentos

Ao professor Sebastián Urrutia, que orientou com prazer e dedicação todas as etapas deste trabalho.
Aos membros da banca examinadora.
À FAPEMIG, pelo incentivo financeiro sem o qual este trabalho não seria possível.
À minha namorada Karoline, pelo amor, carinho e paciência. Você sempre será a minha fonte de inspiração.
Aos meus pais, Luiz e Divina, e ao meu irmão Renato, por todo apoio e carinho fornecidos que me serviram de incentivo a dar continuidade nos estudos até a conclusão deste mestrado.
Meus eternos agradecimentos.

Resumo

A programação de tabelas para competições esportivas é uma área crescente da pesquisa operacional e da ciência da computação. Os problemas de programação de tabelas são de grande interesse econômico, pois a qualidade de uma tabela influencia diretamente no desempenho das equipes e no rendimento financeiro dos patrocinadores e das emissoras de rádio e televisão. A grande importância econômica e o elevado grau de dificuldade de resolução ótima dos problemas têm atraído um número crescente de pesquisadores.

O presente trabalho aborda o Problema do Torneio com Viagens com Estádios Predefinidos. O trabalho é dividido em duas partes. Na primeira, é proposta uma heurística baseada em busca local para obtenção de boas soluções viáveis para o problema. Quatro estruturas de vizinhanças são utilizadas no procedimento de busca local. Um método generalizado baseado em teoria dos grafos é proposto para a geração de soluções iniciais. Na segunda parte, são propostas relaxações combinatórias de um modelo de programação linear inteira que permitem a obtenção de limites duais para o problema.

Os resultados computacionais mostram que tanto a heurística quanto o método de obtenção de limites duais propostos superam os resultados anteriormente conhecidos na literatura para todas as instâncias de teste.

Abstract

Sport timetabling is a growing area of operations research and computer science. Sport timetabling problems have a great economic importance since the quality of the schedule directly influences the performance of teams and the income obtained by the sponsors and broadcasters. This great economic impact and high degree of difficulty of the problems have attracted a growing number of researchers.

This work addresses the Traveling Tournament Problem with Predefined Venues. In the first of this work, a local search based heuristic is proposed to obtain good feasible solutions to the problem. Four neighborhoods are used in the local search procedure. A graph theory based method is proposed to generate initial solutions to the problem. In the second part of this work, a combinatorial relaxation of an integer linear programming is used to obtain dual bounds to the problem.

The computational results show that both the heuristic and the method to generate dual bounds outperform the previous results in the literature for all test instances.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A programação de tabelas para torneios esportivos é uma área crescente da pesquisa operacional e da ciência da computação. Os problemas de programação de tabela são de grande interesse econômico, pois a qualidade de uma tabela influencia diretamente no desempenho das equipes e no rendimento financeiro dos patrocinadores e das emissoras de rádio e televisão. A grande importância econômica e o elevado grau de dificuldade têm atraído um número crescente de pesquisadores.

Inúmeras abordagens têm sido utilizadas para tratar problemas de programação de tabelas. Dentre essas, destacam-se: programação inteira [Nemhauser e Trick, 1998; Melo, 2007; Ribeiro e Urrutia, 2007b; Briskorn, 2006; Croce e Oliveri, 2006; Goossens e Spieksma, 2006; Rasmussen, 2008], programação por restrições [Henz, 1999; Russell e Urban, 2006; Henz et al., 2004], heurísticas de busca local [Anagnostopoulos et al., 2006; Ribeiro e Urrutia, 2007b; Hentenryck e Vergados, 2005; Gaspero e Schaerf, 2007] e métodos híbridos [Benoist et al., 2001; Easton et al., 2003]. Em [Rasmussen e Trick, 2008; Easton et al., 2004; Kendall et al., 2008], são encontradas análises do estado da arte.

Considera-se que seja dado um número n (par) de equipes nos conceitos a seguir. Em um torneio *round robin*, todas as equipes se enfrentam em um número fixo de vezes em um dado número de rodadas. Em um torneio *round robin simples,* cada par de equipes se enfrentam exatamente uma vez. Em um torneio *round robin duplo*, cada par de equipes se enfrentam exatamente duas vezes . Quando o número de rodadas é mínimo e cada equipe joga exatamente uma vez em cada rodada, o torneio é dito ser *compacto*. Considera-se que cada equipe possua o seu estádio localizado em sua cidade sede, e que cada jogo deva ser realizado em um dos estádios das duas equipes participantes. Uma equipe que joga no seu próprio estádio é denominada *mandante,* e a outra equipe que realiza o jogo fora de seu estádio é dita ser a equipe *visitante*. Um escalonamento deve determinar em qual rodada e em qual estádio cada jogo deve ser realizado.

O escalonamento de um torneio *round robin* é usualmente dividido em dois estágios.

Primeiro, a construção do quadro de jogos determina em qual rodada cada jogo se realiza. Segundo, o conjunto de padrões casa-fora (conjunto HAP, do inglês *home-away pattern set*) determina em que condição, em casa ou fora, cada equipe joga em cada rodada. Juntos, o quadro de jogos e o conjunto HAP determinam o escalonamento do torneio.

Alguns problemas da literatura consideram simultaneamente a construção do quadro de jogos e do conjunto HAP. O Problema do Torneio com Viagens (TTP, do inglês *Traveling Tournament Problem*) [Easton et al., 2001] é um problema clássico de escalonamento em esportes. Ele consiste em determinar um quadro de jogos e um conjunto de padrões casa-fora para um torneio *round robin* duplo, tal que nenhuma equipe realize mais do que três jogos consecutivos em casa, nem mais do que três jogos consecutivos fora. Os jogos entre duas equipes quaisquer não podem ocorrer em rodadas consecutivas. O objetivo do problema é minimizar a distância total percorrida pelas equipes.

Em outros problemas, o quadro de jogos ou o conjunto de padrões casa-fora pode estar fixo e ser conhecido previamente. O Problema de Minimização de Quebras [Régin, 2001] e o Problema de Minimização de Distância para um Quadro de Jogos Restrito [Rasmussen e Trick, 2006] são exemplos onde o quadro de jogos é fixo e o objetivo é encontrar um conjunto HAP que otimize uma certa função. O problema de encontrar um quadro de jogos compatível com um conjunto HAP fixo aparece como subproblema em vários métodos para resolver problemas práticos de escalonamentos de competições esportivas [Nemhauser e Trick, 1998; Ribeiro e Urrutia, 2007b].

O problema de escalonar um torneio *round robin* simples pode ser dividido de uma forma diferente. Ao invés de considerar a construção do conjunto HAP, que determina em que condição (em casa ou fora) cada equipe joga em cada rodada, um conjunto de atribuições casa-fora (HAA, do inglês home-away assignment) pode ser considerado. O conjunto de atribuições casa-fora determina, para cada par de equipes, em qual estádio cada partida é realizada. Se o quadro de jogos está fixo, tanto o HAP quanto o HAA fornecem a mesma informação e determinam o escalonamento.

Melo [2007] introduziu a classe de problemas denominada Problemas com Estádios Predefinidos. Os problemas desta classe consideram predeterminados os locais de realização de cada partida, ou seja, o HAA é fixo e conhecido previamente. Um problema dessa classe consiste em escalonar um torneio *round robin* simples, compatível com o HAA, otimizando uma determinada função objetivo e garantindo que nenhuma equipe jogue mais do que três partidas consecutivas em casa nem mais do que três partidas consecutivas fora de casa. Essa classe de problemas é especialmente importante em torneios que são realizados em duas fases, onde os jogos da segunda fase são entre as mesmas equipes da primeira, mas invertendo-se o local da partida. O Problema do Torneio com Viagens com Estádios Predefinidos (PTVEP) [Melo, 2007], objeto deste trabalho, é pertencente a essa classe. Seu objetivo

consiste em minimizar a distância total percorrida pelas equipes. O presente trabalho aborda o Problema do Torneio com Viagens com Estádios Predefinidos.

Instâncias com até 8 equipes do PTVEP foram resolvidas usando modelos de programação linear inteira em [Melo, 2007]. Uma vez que o resolvedor comercial não foi capaz de encontrar soluções viáveis para instâncias com dezoito ou mais equipes, quatro heurísticas baseadas nas formulações de programação linear inteira foram também desenvolvidas em [Melo, 2007]. O presente trabalho propõe uma heurística baseada em busca local para o PTVEP, visando encontrar boas soluções viáveis para instâncias de tamanhos reais (acima de 18 equipes). Este propõe ainda uma relaxação combinatória para um dos modelos de programação linear inteira de [Melo, 2007] baseada nos limites inferiores independentes de [Easton et al., 2001], capaz de melhorar os limites duais de relaxação linear apresentados em [Melo, 2007].

O texto está organizado da seguinte forma:

No capítulo 2, são introduzidas as notações e os conceitos utilizados ao longo do texto. Em seguida, o PTVEP é formulado como um problema de otimização em grafos.

No capítulo 3, propõe-se uma heurística baseada na metaheurística de Busca Local Iterada para obter boas soluções viáveis para instâncias de tamanhos de interesse prático. Apresenta-se cada elemento presente na heurística: procedimento construtivo, vizinhanças, busca local e perturbações. Finalmente, são apresentados os resultados computacionais obtidos.

No capítulo 4, são propostos novos métodos para obtenção de limites duais para o PTVEP baseados na relaxação de um dos modelos de programação linear inteira de [Melo, 2007] e nos limites inferiores independentes de [Easton et al., 2001]. Em seguida, apresentam-se os resultados computacionais.

No capítulo 5, resumem-se as principais contribuições deste trabalho e são sugeridas novas possibilidades de trabalhos futuros.

Capítulo 2

Formalização

2.1 Introdução

Em um PTVEP são dados: (i) Um conjunto com n (par) equipes $\{1,\ldots,n\}$, (ii) uma matriz $D(n\times n)$ de distâncias (não necessariamente simétrica), onde cada elemento d_{ij} de D contém a distância entre as cidades sedes das equipes i e j e (iii) um conjunto de atribuições casa-fora $J=\{(i,j)$: o jogo entre i e j ocorre na casa de $i\}$. O conjunto J determina o local de realização de cada partida do torneio. Se $(i,j)\in J$, então a partida entre i e j ocorre na casa de i, caso contrário, a partida entre i e j ocorre na casa de j, ou seja, $(j,i)\in J$. O objetivo do problema é escalonar um torneio *round robin* simples e compacto que seja compatível com J, minimizando a distância total percorrida pelas equipes, sujeito à restrição de que nenhuma equipe realize mais do que três partidas em rodadas consecutivas em casa, nem mais do que três partidas em rodadas consecutivas fora de casa. As equipes devem iniciar e terminar a competição em suas respectivas cidades sedes. Toda vez que uma equipe realiza duas partidas em rodadas consecutivas fora de casa, ela viaja diretamente da cidade do primeiro adversário para a cidade do próximo adversário. Segue um exemplo de uma instância com seis equipes.

Exemplo 2.1.1. *(i) O conjunto de equipes é* $\{1,2,3,4,5,6\}$*, (ii) a matriz de distâncias,*

$$D=\begin{pmatrix} 0 & 1 & 2 & 3 & 2 & 1\\ 1 & 0 & 1 & 2 & 3 & 2\\ 2 & 1 & 0 & 1 & 2 & 3\\ 3 & 2 & 1 & 0 & 1 & 2\\ 2 & 3 & 2 & 1 & 0 & 1\\ 1 & 2 & 3 & 2 & 1 & 0 \end{pmatrix}$$

e (iii) o HAA é dado por

$$\begin{aligned} J = \{ \quad & (1,2),(1,4),(2,4),(2,6),(3,1), \\ & (3,2),(3,6),(4,3),(4,5),(4,6), \\ & (5,1),(5,2),(5,3),(6,1),(6,5)\}. \end{aligned}$$

Uma solução ótima para esse problema pode ser dada pela tabela de jogos:

Equipes	**Rodadas**				
	1	**2**	**3**	**4**	**5**
1	-3	2	-5	-6	4
2	-5	-1	6	4	-3
3	1	6	-4	-5	2
4	6	5	3	-2	-1
5	2	-4	1	3	-6
6	-4	-3	-2	1	5

A linha i da tabela contém a sequência de partidas que a equipe i deve realizar. O sinal negativo indica a ocorrência da partida na casa do adversário de i e a ausência de sinal indica a ocorrência da partida na casa de i. Considere como exemplo a equipe 3. Ela realiza suas duas primeiras partidas em casa, contra as equipes 1 e 6. Em seguida, ela realiza duas partidas consecutivas fora de casa, viajando de sua cidade para a cidade da equipe 4 . Da cidade da equipe 4, parte diretamente para a cidade da equipe 5. Finalmente, viaja para sua cidade sede para realizar a sua última partida em casa, contra a equipe 2. O custo relacionado às viagens da equipe 3 é $d_{34} + d_{45} + d_{53} = 1 + 1 + 2 = 4$. O custo total da solução apresentada é 36.

Uma instância do PTVEP é dita ser balanceada quando a quantidade de partidas que cada equipe realiza em casa é $n/2$ ou $n/2-1$. Caso contrário, a instância e dita ser desbalanceada. Melo [2007] observou que o resolvedor de programação linear inteira encontra rapidamente soluções viáveis para instâncias balanceadas. Para as instâncias não balanceadas, as restrições que limitam a quantidade de partidas consecutivas em casa ou fora de casa, para cada equipe, são mais difíceis de serem atendidas, dificultando assim a busca por soluções viáveis. Ainda não se conhece um algoritmo polinomial que encontre uma solução viável (ou detecte inviabilidade) para o PTVEP.

2.2 Fatoração de grafos

Um grafo é um par ordenado (V,E) onde V é um conjunto finito de vértices e E é um conjunto finito de arestas, com V e E disjuntos. Em um grafo orientado, cada aresta conecta um par

ordenado de vértices. Em um grafo não-orientado, cada aresta conecta um par de vértices, não importando a ordem. Um laço é uma aresta que conecta um vértice v ao próprio vértice v. Um grafo simples não possui laços e, além disso, possui no máximo uma aresta conectando o mesmo par de vértices. Os grafos tratados neste texto serão sempre grafos simples e não-orientados. A aresta $e \in E$ que conecta dois vértices distintos v_1 e v_2 (aresta com uma extremidade em v_1 e a outra extremidade em v_2) será denotada por (v_1, v_2) ou (v_2, v_1), sem que haja possibilidade de confusão. O grau $\mu(v)$ de um vértice v é a quantidade de arestas com extremidade em v.

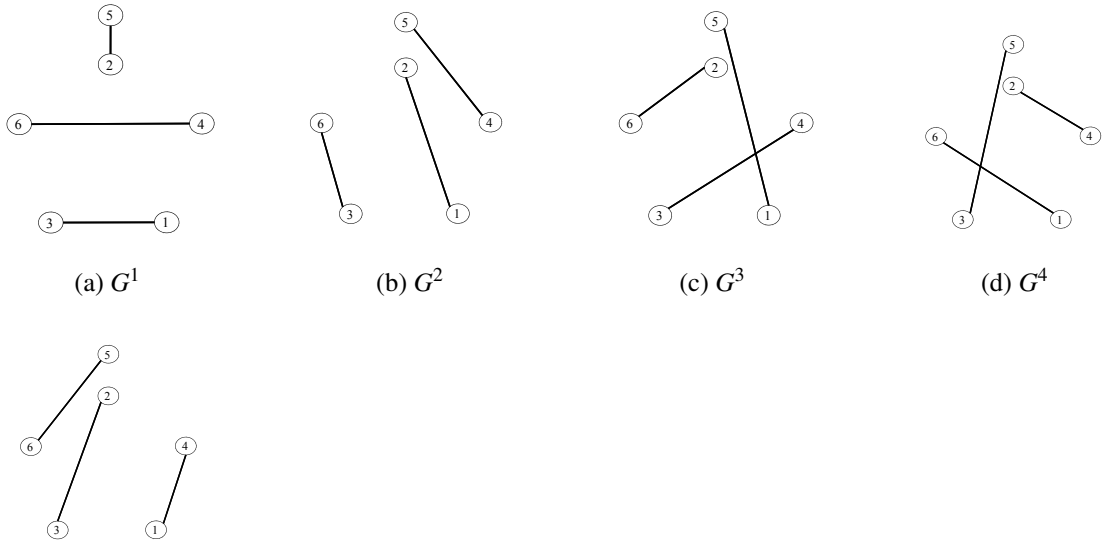
Um grafo é completo se para todo par de vértices distintos existe uma aresta. O grafo completo com conjunto de vértices $\{1,...,n\}$ será denotado por K_n. Uma partição de um conjunto X qualquer é uma coleção de conjuntos $\{X_1, X_2, ..., X_p\}$ disjuntos dois a dois e não vazios, tal que $\cup_{i=1}^{p} X_i = X$. Um grafo é bipartido do tipo (p,q) se existe uma partição do conjunto de vértices $\{V_1, V_2\}$ com cardinalidades $|V_1| = p$ e $|V_2| = q$, tal que nenhuma aresta tenha ambas as extremidades em V_1 nem ambas as extremidades em V_2. Um grafo bipartido completo, denotado por $(V = V_1 + V_2, E)$, é um grafo com partição dos vértices $\{V_1, V_2\}$ e conjunto de arestas $E = \{\{(u,v) : u \in V_1, v \in V_2\}$.

Considere o grafo $G = (V,E)$. Um subgrafo de G é um grafo $G^* = (V^*, E^*)$, tal que $V^* \subseteq V$ e $E^* \subseteq E$. Um fator de G é um subgrafo $G' = (V,E')$ com $E' \subseteq E$. G' é dito ser um 1-fator se todos os seus vértices possuem grau um. Uma fatoração de G é um conjunto de fatores $\{G^1 = (V,E^1), \ldots, G^p = (V,E^p)\}$, tal que $\{E_1, \ldots, E_p\}$ é uma partição de E. Uma 1-fatoração de G é uma fatoração com todos os fatores sendo 1-fatores. Em uma 1-fatoração ordenada de G, os 1-fatores são considerados em uma determinada ordem.

Conceitos de teoria dos grafos não explicados aqui podem ser consultados, por exemplo, em [Bondy e Murty, 1979].

Um escalonamento de um torneio *round robin* simples com n (par) equipes e conjunto de atribuições casa-fora predefinido pode ser representado por uma 1-fatoração ordenada $\{G^1 = (V,E^1), \ldots, G^p = (V,E^{n-1})\}$ de K_n [Ribeiro e Urrutia, 2007a]. Cada vértice do grafo representa uma equipe. Cada aresta $(i,j) \in E^l$ representa a ocorrência da partida entre as equipes i e j na l-ésima rodada na cidade dada pelo conjunto de atribuições casa-fora. A Figura 2.1 mostra um exemplo de uma 1-fatoração ordenada para K_6. Essa 1-fatoração ordenada representa o mesmo escalonamento apresentado no exemplo 2.1.1.

Duas 1-fatorações F e H do mesmo grafo $G = (V,E)$ são isomorfas se existe uma função bijetiva $\alpha : V \to V$, tal que a aplicação de α a todos os vértices de todos os 1-fatores de F resulte em H. A quantidade $N(n)$ de 1-fatorações não-isomorfas de K_n (n par) cresce muito rapidamente. Segundo Dinitz et al. [1994], $N(2) = N(4) = N(6) = 1$, $N(8) = 6$, $N(10) = 396$, $N(12) = 526915620$. Estima-se que $N(14) \approx 1,132 \times 10^{18}$, $N(16) \approx 7,07 \times 10^{30}$ e que $N(18) \approx 2,374 \times 10^{47}$.



(a) G^1 (b) G^2 (c) G^3 (d) G^4

(e) G^5

Figura 2.1: Exemplo de uma 1-fatoração para k_6.

2.3 Formalização do PTVEP em grafos

Dados: (i) um conjunto com n (par) equipes $V = \{1,\ldots,n\}$; (ii) uma matriz $D(n \times n)$ de distâncias, não necessariamente simétrica, com os elementos da diagonal nulos, onde cada elemento $D[i,j]$ contém a distância entre as cidades sedes das equipes i e j; (iii) um conjunto de atribuições casa-fora $J = \{(i,j) : \text{o jogo entre } i \text{ e } j \text{ ocorre na casa de } i\}$ e (iv) uma 1-fatoração ordenada $F = (f_1 = (V,A_1),...,f_{n-1} = (V,A_{n-1}))$ de K_n, definem-se as funções a seguir.

A função

$$adv_F(v,r) = u, \text{ para o único } u \in V \text{ tal que } (u,v) \in A_r \tag{2.1}$$

retorna o único vértice u adjacente a v em f_r. O vértice u retornado representa a equipe que é adversária de v na rodada r. O subscrito F denota a 1-fatoração correspondente.

A função

$$mandante(u,v) = \begin{cases} u, & \text{Se } (u,v) \in J \\ v, & \text{caso contrário} \end{cases} \tag{2.2}$$

retorna o vértice associado à equipe que joga em casa no confronto entre as duas equipes u e v.

As funções

$$\begin{aligned} &dc(v) = D[v, mandante(v, adv_F(v,1)] \\ &\text{e} \\ &df(v) = D[v, mandante(v, adv_F(v,n-1)] \end{aligned} \tag{2.3}$$

retornam, respectivamente, as distâncias viajadas pela equipe associada ao vértice v antes de realizar a primeira partida e a distância para voltar à cidade sede após a última rodada.

A função

$$di(t,u,v) = D[mandante(t,u), mandante(t,v)] \tag{2.4}$$

retorna a distância que a equipe t deve viajar para enfrentar a equipe v logo após ter enfrentado a equipe u.

Então, a função

$$dt(v) = dc(v) + \sum_{r=1}^{n-2} [di(v, adv(v,r), adv(v,r+1))] + df(v) \tag{2.5}$$

fornece a distância viajada pela equipe associada ao vértice v durante o torneio. O primeiro termo considera a distância viajada para realizar a primeira partida. O somatório considera as distâncias percorridas para realizar as partidas da segunda até a última rodada. O último termo considera a distância percorrida para retornar à cidade sede após a última rodada.

Visto como um problema em grafos, o PTVEP consiste em obter uma 1-fatoração ordenada de K_n que minimize a soma das distâncias viajadas pelas equipes durante o torneio, $\sum_{v \in V} dt(v)$, sujeito à

$$1 \leq |\{mandante(u, adv_F(u,i)) : i = j, ..., j+3\} \backslash \{u\}| \leq 3, \forall u \in V \text{ e } j = 1, ..., n-4. \tag{2.6}$$

A restrição garante que nenhuma equipe u realize mais do que três partidas consecutivas em casa, nem mais do que três partidas consecutivas fora de casa.

Capítulo 3

Uma Heurística de Busca Local para o PTVEP

Uma classe de problemas de otimização, denominada NP-difícil, possui vários problemas relacionados que, até a presente data, não se conhecem algoritmos que os resolvam com complexidade de tempo polinomial com relação ao tamanho da entrada. Os melhores algoritmos conhecidos para resolvê-los possuem complexidade de tempo exponencial. Isso implica que, a partir de um determinado tamanho da entrada, um algoritmo exato não consegue mais fornecer uma solução ótima em um tempo razoável. Devido a esta limitação, para muitos problemas dessa classe, instâncias de interesse prático não podem ser resolvidas na otimalidade. Nesse contexto, surgem as *heurísticas*, que são algoritmos que se propõem a encontrar boas soluções viáveis com baixo custo computacional. Em contrapartida, as heurísticas não oferecem garantia alguma de qualidade.

Embora ainda não se conheça uma prova formal, acredita-se que o PTVEP seja NP-difícil. Em [Melo, 2007], instâncias com até 8 equipes foram resolvidas rapidamente (poucos segundos), enquanto que nenhuma instância com dez ou mais equipes pôde ser resolvida na otimalidade dentro de duas horas de execução. Nesse contexto, justifica-se o desenvolvimento de heurísticas para tentar encontrar boas soluções viáveis para instâncias com tamanhos de interesse prático para o PTVEP.

Neste capítulo, descreve-se a heurística proposta para o PTVEP. Inicialmente, na seção 3.1, definem-se conceitos básicos sobre o desenvolvimento de heurísticas e apresenta-se a metaheurística Busca Local Iterada. Em seguida, nas seções 3.2-3.5, apresenta-se cada elemento da heurística proposta. Finalmente, na última seção, apresentam-se os resultados computacionais.

3.1 Heurísticas

Uma heurística construtiva é aquela que usa apenas a instância para tentar construir uma solução viável para o problema, enquanto uma heurística de melhoria tem o papel de melhorar uma dada solução construída. Muitas heurísticas de melhoria são baseadas no conceito de vizinhança. Uma *vizinhança* $N(s)$ de uma determinada solução s para um problema de otimização é definida como um conjunto de soluções que podem ser atingidas efetuando-se um determinado tipo de operação sobre s. As operações utilizadas, em geral, provocam pequenas modificações nos elementos da solução. Uma vizinhança pode ser definida a partir de outras por meio de operações sobre conjuntos. Uma vizinhança é dita ser *conexa* quando dadas duas soluções quaisquer s_1 e s_2 do espaço de busca, é possível partir de s_1 e chegar a s_2, aplicando-se uma sequência finita de movimentos na vizinhança considerada.

Uma heurística de busca local é uma heurística de melhoria que parte de uma solução inicial qualquer e procura sistematicamente por uma solução que seja melhor do que a solução corrente em uma vizinhança considerada. Se uma solução melhor do que a solução corrente é encontrada, a nova solução passa a ser a solução corrente e o algoritmo continua em busca de melhores soluções vizinhas. Se não existe uma solução vizinha melhor do que a solução corrente, o algoritmo termina e a solução corrente, denominada ótimo local na vizinhança considerada, é retornada. Mais informações sobre heurísticas de busca local podem ser encontradas em [Hoos e Stuzle, 2005], por exemplo.

Uma metaheurística é um padrão para desenvolvimento de heurísticas. Ela define um conjunto de regras para criação de heurísticas, que não são específicas a nenhum problema em particular. Uma metaheurística pode ser utilizada para criar heurísticas para problemas completamente diferentes. Pode-se também utilizar uma mesma metaheurística para criar diversas heurísticas para um mesmo problema, diferentes em abordagem e/ou eficiência. Em [Glover e Kochenberger, 2003], encontram-se descrições sobre as metaheurísticas mais utilizadas na literatura.

A metaheurística de Busca Local Iterada (ILS, do inglês, Iterated Local Search) [Lourenço et al., 2003] propõe o uso de perturbações (pequenas modificações na solução) para escapar de ótimos locais. O algoritmo 1 resume os passos dessa metaheurística. O método se inicia pela construção de uma solução inicial (linha 1) e a aplicação de uma busca local sobre a solução construída (linha 2). A cada iteração, a solução corrente é perturbada (linha 4), e um método de busca local é aplicado sobre a solução perturbada (linha 5). Ao final de cada iteração, um critério de aceitação é avaliado para decidir se a solução obtida deve ou não substituir a solução corrente (linha 6). O procedimento (linhas 4-6) é repetido até que uma condição de término seja atendida.

As próximas seções deste capítulo descrevem os elementos que compõem a heurística baseada em Busca Local Iterada proposta para o PTVEP.

Algoritmo 1: Busca Local Iterada

```
1 s_0 ← GeraSolucaoInicial ;
2 s* ← BuscaLocal(s_0) ;
3 enquanto condição de término não satisfeita faça
4 |   s' ← Perturbacao(s*, histria) ;
5 |   s*' ← BuscaLocal(s') ;
6 |   s* ← CriterioAceitacao(s*, s*', historia) ;
7 fim
```

3.2 Vizinhanças para o PTVEP

O espaço de busca considerado pela heurística descrita para uma instância qualquer com n equipes considera todas as 1-fatorações ordenadas de K_n, incluindo aquelas que violam a restrição (2.6) de que nenhuma equipe pode realizar mais do que três partidas consecutivas fora de casa, nem mais do que três partidas consecutivas em casa.

Diferentes estruturas de vizinhanças têm sido utilizadas em procedimentos de busca local para escalonamento de torneios *round robin*, por exemplo, [Anagnostopoulos et al., 2006; Gaspero e Schaerf, 2007; Ribeiro e Urrutia, 2007a]. O presente trabalho considera quatro delas no contexto do PTVEP. Seja $X = \{F^1, ..., F^{n-1}\}$ uma 1-fatoração ordenada para o grafo K_n com conjunto de vértices $V = \{1, \ldots, n\}$ (n par). Denote por $R = \{1, ..., n-1\}$ o conjunto de índices de X. Denote por $advs_X(i,U)$ o conjunto de todos os vértices adjacentes a i no subconjunto de índices $U \subseteq R$, ou seja, $advs_X(i,U) = \{adv_X(i,j) : j \in U\}$.

A primeira vizinhança é a *troca de equipes* ($N1$). Dados $t_1, t_2 \in V$, com $t_1 \neq t_2$, a 1-fatoração ordenada obtida trocando-se todos os vértices adjacentes a t_1 e t_2 em todos os fatores, com exceção do fator que possui a aresta (t_1, t_2), é vizinha de X segundo $N1$.

A segunda vizinhança é a *troca de rodadas* ($N2$). Dados $r_1, r_2 \in R$, com $r_1 \neq r_2$, a 1-fatoração ordenada obtida trocando-se todas as arestas entre os fatores F^{r_1} e F^{r_2} é vizinha de X segundo $N2$. O movimento equivale a trocar a ordem entre os dois fatores.

A terceira vizinhança é a *troca parcial de equipes* ($N3$). Dados $t_1, t_2 \in V$ e $r \in R$, com $t_1 \neq t_2$ e $adv_X(t_1, r) \neq t_2$. Seja S o subconjunto de R de cardinalidade mínima, tal que $r \in S$ e o conjunto de vértices adjacentes a t_1 e t_2 nos fatores indexados pelos elementos de S sejam os mesmos, ou seja, $S \subseteq R$ de cardinalidade mínima, tal que $r \in S$ e $advs_X(t_1, S) = advs_X(t_2, S)$. A 1-fatoração ordenada obtida trocando-se os vértices adjacentes a t_1 e t_2 em todos os fatores indexados pelos elementos de S é vizinha de X segundo $N3$.

A Figura 3.1 ilustra um movimento na vizinhança $N3$ para um torneio com oito equipes e $r = 2$, $t_1 = 1$ e $t_2 = 2$. O conjunto S que determina o movimento pode ser calculado passo-a-passo da seguinte forma: Inicialmente, $S_1 = \{2\}$, $advs_X(t_1, S_1) = \{8\}$ e $advs_X(t_2, S_1) = \{5\}$. Como $advs_X(t_1, S_1) \neq advs_X(t_2, S_1)$, deve-se inserir em S_1 a rodada

em que t_1 enfrenta 5 (rodada 6) e também a rodada em que t_2 enfrenta 8 (rodada 5). No segundo passo, $S_2 = \{2,5,6\}$, $advs_X(t_1,S_1) = \{3,5,8\}$ e $advs_X(t_2,S_1) = \{5,7,8\}$. Como $advs_X(t_1,S_2) \neq advs_X(t_2,S_2)$, deve-se inserir em S_2 a rodada em que t_1 enfrenta 7 (rodada 7) e também a rodada em que t_2 enfrenta 3 (rodada 7). Finalmente, $S_3 = \{2,5,6,7\}$ e $advs_X(t_1,S_2) = advs_X(t_2,S_2) = \{3,5,7,8\}$.

Times	Rodadas						
	1	$r=2$	3	4	5	6	7
$t_1 = 1$	-4	8	6	-2	-3	5	7
$t_2 = 2$	-6	-5	4	1	-8	-7	3
3	5	-4	-7	-6	1	8	-2
4	1	3	-2	-8	7	-6	-5
5	-3	2	-8	-7	6	-1	4
6	2	-7	-1	3	-5	4	8
7	-8	6	3	5	-4	2	-1
8	7	-1	5	4	2	-3	-6

(a) Tabela original X

Times	Rodadas						
	1	$r=2$	3	4	5	6	7
$t_1 = 1$	-4	5	6	-2	8	7	-3
$t_2 = 2$	-6	-8	4	1	3	-5	-7
3	5	-4	-7	-6	-2	8	1
4	1	3	-2	-8	7	-6	-5
5	-3	-1	-8	-7	6	2	4
6	2	-7	-1	3	-5	4	8
7	-8	6	3	5	-4	-1	2
8	7	2	5	4	-1	-3	-6

(b) Nova tabela obtida após o movimento em N_3

Figura 3.1: Movimento na vizinhança N_3 para um torneio com 8 equipes e $r = 2$, $t_1 = 1$, e $t_2 = 2$ (as entradas destacadas em (a) aparecem modificadas em (b) após o movimento).

A quarta vizinhança é a *troca parcial de rodadas* ($N4$). Dados $r_1, r_2 \in R$, com $r_1 \neq r_2$, e $t \in V$. Seja U_t o subconjunto de V de cardinalidade mínima, tal que $t \in U_t$ e os vértices adjacentes aos vértices em U_t no fator F^{r_1} sejam os mesmos vértices adjacentes aos vértices em U_t no fator F^{r_2}, ou seja, $U_t \subseteq V$ de cardinalidade mínima, tal que $t \in U_t$ e $\{adv_X(y,r_1) : y \in U_t\} = \{adv_X(y,r_2) : y \in U_t\}$. A 1-fatoração ordenada obtida trocando-se as arestas $(y, adv_X(y,r_1))$ e $(y, adv_X(y,r_2))$ entre os 1-fatores r_1 e r_2 para todo $y \in U_t$ é vizinha de X segundo $N4$.

A Figura 3.2 ilustra um movimento na vizinhança $N4$ para um torneio com dez equipes e $t_1 = 1$, $r_1 = 1$ e $r_2 = 4$. Nesse caso, $U_t = \{1,6,7\}$. As equipes 3, 4 e 8 são as adversárias

das equipes em U_t nas rodadas 1 e 4.

Times	Rodadas								
	$r_1 = 1$	2	3	$r_2 = 4$	5	6	7	8	9
$t = 1$	-3	10	7	-8	-5	2	4	9	-6
2	9	8	6	-5	-3	-1	10	-4	-7
3	1	-4	9	-6	2	-10	-7	-8	5
4	-6	3	-10	-7	8	5	-1	2	9
5	10	7	8	2	1	-4	-9	-6	-3
6	4	-9	-2	3	-10	-7	-8	5	1
7	-8	-5	-1	4	-9	6	3	-10	2
8	7	-2	-5	1	-4	-9	6	3	-10
9	-2	6	-3	10	7	8	5	-1	-4
10	-5	-1	4	-9	6	3	-2	7	8

(a) Tabela Original X

	Rodadas								
Times	$r_1 = 1$	2	3	$r_2 = 4$	5	6	7	8	9
$t = 1$	-8	10	7	-3	-5	2	4	9	-6
2	9	8	6	-5	-3	-1	10	-4	-7
3	-6	-4	9	1	2	-10	-7	-8	5
4	-7	3	-10	-6	8	5	-1	2	9
5	10	7	8	2	1	-4	-9	-6	-3
6	3	-9	-2	4	-10	-7	-8	5	1
7	4	-5	-1	-8	-9	6	3	-10	2
8	1	-2	-5	7	-4	-9	6	3	-10
9	-2	6	-3	10	7	8	5	-1	-4
10	-5	-1	4	-9	6	3	-2	7	8

(b) Nova tabela obtida após movimento em N_4

Figura 3.2: Movimento em N_4 para um torneio com dez equipes e $t = 1$, $r_1 = 1$, e $r_2 = 4$ (as entradas destacadas em (a) aparecem modificadas em (b) após o movimento).

Movimentos em $N1 \cup N2$ não permitem mudar a estrutura da 1-fatoração corrente. Todas as 1-fatorações ordenadas que podem ser obtidas por meio de movimentos em $N1$ ou $N2$ geram sempre 1-fatorações ordenadas que diferem apenas na ordem dos fatores ou são isomorfas entre si. Apenas movimentos em $N3$ ou $N4$ podem mudar a estrutura da 1-fatoração corrente. Para certos valores de n, pode existir uma 1-fatoração F de K_n tal que $N1(F) = N3(F)$ e $N2(F) = N4(F)$, ou seja, cada movimento em $N3$ (resp. $N4$), a partir de F, é equivalente a algum movimento em $N1$ (resp. $N2$). Quando esse fenômeno acontece, a heurística fica presa na estrutura da 1-fatoração corrente e não consegue explorar outras porções do espaço de busca, o que significa que a vizinhança $N_1 \cup N_2 \cup N_3 \cup N_4$ pode não ser conexa, dependendo do valor de n. Verificou-se, experimentalmente, que esse fenômeno acontece, por exemplo, para $n \in \{12, 14, 20\}$ e 1-fatoração canônica (veja a definição 3.3.1).

Pode-se reproduzir qualquer movimento de $N2$ por uma sequência finita de um ou mais movimentos em $N4$, como se demonstra a seguir.

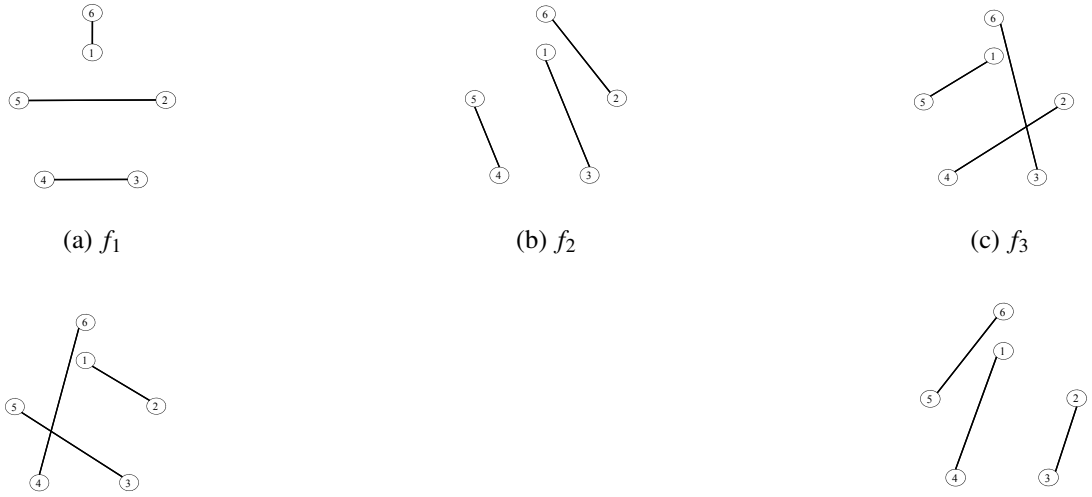
Um movimento em $N2$ com parâmetros r_1 e r_2 equivale a trocar todas as arestas entre os fatores F^{r_1} e F^{r_2}. Um movimento em $N4$ com parâmetros t, r_1 e r_2 equivale a trocar apenas um subconjunto de arestas entre F^{r_1} e F^{r_2}. O conjunto de arestas envolvidas são aquelas incidentes aos vértices de U_t, calculado durante o movimento. Então, se $t' \in U_t$ ou t' é adjacente a algum elemento de U_t em F^{r_1} ou F^{r_2} , o movimento com parâmetros t', r_1 e r_2 é equivalente ao movimento com parâmetros t, r_1 e r_2. Por outro lado, se $t' \notin U_t$, e ainda, t' não é adjacente a nenhum elemento de U_t, então o movimento com parâmetros t', r_1 e r_2 afeta um subconjunto de arestas disjunto do conjunto de arestas afetadas pelo movimento com parâmetros t, r_1 e r_2. O conjunto de vértices pode, então, ser particionado em $\{P_1, \ldots, P_p\}$, de tal forma que para $i = 1, \ldots, p$, os movimentos em $N4$ com parâmetros t, r_1, r_2 são equivalentes para todo $t \in P_i$. Dessa forma, conclui-se que qualquer sequência de p movimentos com parâmetros r_1, r_2 e uma equipe de cada um dos subconjuntos P_i, $i = i, \ldots p$, é equivalente a um movimento em $N2$ com parâmetros r_1 e r_2.

Diz-se, então, que a vizinhança $N4$ generaliza a vizinhança $N2$. De forma semelhante, pode-se também mostrar que a vizinhança $N3$ generaliza a vizinhança $N1$.

3.3 Solução Inicial

Uma 1-fatoração de K_n, denominada 1-fatoração canônica [de Werra, 1981], tem sido muito utilizada na construção de escalonamentos iniciais para torneios *round robin.* Esta seção descreve um método que generaliza a construção da 1-fatoração canônica para K_n, de forma a obter uma variedade maior de soluções iniciais. O método consiste basicamente em particionar o grafo original em três subgrafos: dois subgrafos completos com $n/2$ vértices cada e um terceiro subgrafo bipartido completo com n vértices. Uma 1-fatoração para o grafo original K_n pode, então, ser obtida por meio da combinação de 1-fatorações para os três subgrafos de K_n. As 1-fatorações para os subgrafos completos com $n/2$ vértices possuem $n/2-1$ fatores cada, e a 1-fatoração para o subgrafo bipartido completo $K_{n/2,n/2}$ possui $n/2$ fatores. A combinação das três 1-fatorações resulta nos $n-1$ fatores da 1-fatoração construída para K_n. A 1-fatoração que deve ser obtida para cada um dos subgrafos de K_n pode ser canônica, ou pode ser construída pela aplicação recursiva do método.

As próximas duas subseções descrevem detalhadamente o método construtivo para obter 1-fatorações para grafos completos e grafos bipartidos completos, por meio do particionamento do grafo original. A terceira subseção formaliza um algoritmo para construção de um conjunto de soluções iniciais para o PTVEP.



(a) f_1 (b) f_2 (c) f_3

(d) f_4 (e) f_5

Figura 3.3: 1-fatoração canônica de K_6

3.3.1 Construção de 1-fatorações para Grafos Completos

A 1-fatoração canônica para o grafo completo K_n é definida como segue:

Definição 3.3.1. *A 1-fatoração canônica [de Werra, 1980]* $F = (f_1 = (\{1,...,n\},A_1),\ldots,f_{n-1} = (\{1,...,n\},A_{n-1})\}$ *de* K_n *satisfaz que para* $i = 1,...,n-1$,

$$A_i = \{(n,i)\} \cup \{(g_1(i,k),g_2(i,k)) : k = 1,\ldots,n/2-1\}, \tag{3.1}$$

com

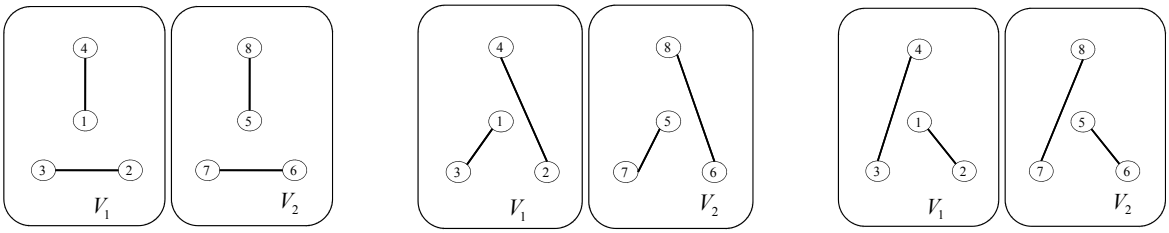
$$g_1(i,k) = \begin{cases} i+k & Se\ i+k < n \\ i+k-n+1 & Se\ i+k \geq n \end{cases}$$

e

$$g_2(i,k) = \begin{cases} i-k & Se\ i-k > 0 \\ i-k+n-1 & Se\ i-k \leq 0 \end{cases}$$

A Figura 3.3 ilustra todos os fatores da 1-fatoração canônica de K_6.

Este trabalho propõe um método generalizado para construção de 1-fatorações para K_n. Suponha inicialmente n um múltiplo de quatro. O método consiste em particionar o conjunto de vértices em dois subconjuntos, V_1 e V_2, com $n/2$ elementos cada. Uma 1-fatoração é construída para os grafos completos induzidos por cada um dos subconjuntos de vértices V_1 e V_2. Para construir os primeiros $n/2-1$ fatores da 1-fatoração de K_n, é feita a união de algum fator da 1-fatoração associada ao subconjunto V_1 com algum fator da 1-fatoração associada ao subconjunto V_2. O processo é repetido usando pares de fatores não utilizados de



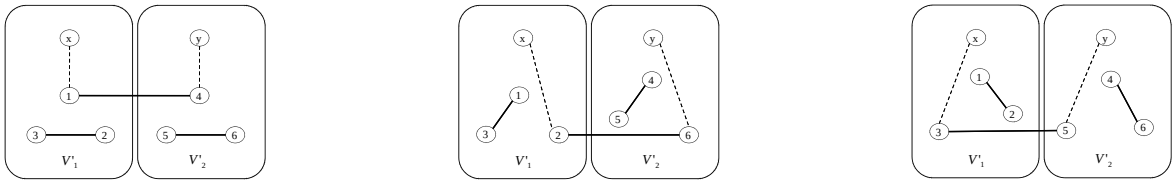
(a) f_1 (b) f_2 (c) f_3

Figura 3.4: As Figuras (a)-(c) exibem os três primeiros 1-fatores obtidos por meio da aplicação do método generalizado para o grafo K_8. O conjunto de vértices foi particionado em $V_1 = \{1,2,3,4\}$ e $V_2 = \{5,6,7,8\}$. A 1-fatoração canônica foi utilizada em cada um dos subconjuntos.

1-fatorações associadas aos subconjuntos V_1 e V_2 até que $n/2-1$ fatores da 1-fatoração de K_n sejam obtidos. Obter os $n/2$ fatores restantes corresponde a obter uma 1-fatoração para o grafo bipartido completo $K_{n/2,n/2}$, definido pela partição $\{V_1, V_2\}$. A seção 3.3.2 apresenta métodos para geração de 1-fatorações para $K_{n/2,n/2}$.

A Figura 3.4 ilustra o uso do método na construção dos $n/2-1=3$ primeiros fatores de K_8, com conjunto de vértices particionado em $V_1 = \{1,2,3,4\}$ e $V_2 = \{5,6,7,8\}$, usando 1-fatoração canônica em cada um dos subconjuntos.

Quando n (par) não é múltiplo de quatro, V é particionado em dois conjuntos V_1 e V_2 com $n/2$ elementos cada. Dois vértices artificiais x e y são incluídos em V_1 e V_2, respectivamente, para formar $V_1' = V_1 \cup \{x\}$ e $V_2' = V_2 \cup \{y\}$, cada um com um número par de vértices. Uma 1-fatoração é construída para os grafos completos definidos por cada um dos conjuntos de vértices V_1' e V_2'. Para construir os primeiros $n/2$ fatores da 1-fatoração de k_n, é feita uma combinação de algum fator da 1-fatoração associada ao conjunto V_1' com algum fator da 1-fatoração associada ao conjunto V_2'. Essa combinação de fatores considera todas as arestas não incidentes aos vértices artificiais e uma nova aresta (v_1, v_2) formada pelos vértices v_1 adjacentes a x, no 1-fator associado ao conjunto V_1', e v_2 adjacente a y, no 1-fator associado ao conjunto V_2'. O processo é repetido usando pares de fatores não utilizados de 1-fatorações associadas aos subconjuntos V_1' e V_2' até que $n/2$ fatores da 1-fatoração de K_n sejam obtidos. Os $n/2-1$ fatores da 1-fatoração de K_n restantes podem ser obtidos por meio de uma 1-fatoração para o grafo bipartido completo $K_{n/2,n/2}$, definido pela partição $\{V_1, V_2\}$. A 1-fatoração do grafo bipartido completo deve possuir um 1-fator que inclua todas as arestas com uma extremidade em V_1 e a outra extremidade em V_2 que já foram inseridas nos primeiros $n/2$ fatores de K_n. Caso contrário, pode-se modificar a 1-fatoração obtida para $K_{n/2,n/2}$ por meio de uma bijeção sobre o conjunto de vértices para que ela possua o 1-fator predeterminado. Esse 1-fator é descartado e os demais compõem os $n/2-1$ fatores restantes da 1-fatoração de K_n.



(a) f_1 (b) f_2 (c) f_3

Figura 3.5: As Figuras (a)-(c) exibem os três primeiros 1-fatores obtidos por meio da aplicação do método generalizado para o grafo K_6. O conjunto de vértices foi particionado em $V_1 = \{1,2,3\}$ e $V_2 = \{4,5,6\}$. A 1-fatoração canônica foi utilizada em cada um dos subconjuntos.

A Figura 3.5 ilustra o uso do método na construção dos $n/2 = 3$ primeiros fatores de K_6, com o conjunto de vértices particionado em $V_1 = \{1,2,3\}$, $V_2 = \{4,5,6\}$ e 1-fatoração canônica utilizada em cada um dos subconjuntos. As arestas tracejadas correspondem às arestas incidentes aos vértices artificiais e não fazem parte dos fatores de K_6.

.

3.3.2 Construção de 1-fatorações para Grafos Bipartidos Completos

Uma 1-fatoração denominada 1-fatoração canônica para um grafo bipartido completo do tipo $(n/2, n/2)$ pode ser obtida fazendo-se um cruzamento alternado de forma circular entre os vértices dos dois subconjuntos que definem a partição (veja [de Werra, 1980]). Ela é definida como se segue.

Definição 3.3.2. *Seja $K_{n/2,n/2}$ definido pela partição $\{V_1 = \{a_1,\ldots,a_{n/2}\}, V_2 = \{b_1,\ldots,b_{n/2}\}\}$. A 1-fatoração canônica de $K_{n/2,n/2}$ é definida como $\{G^1 = (V,E^1),\ldots,G^{n/2} = (V,E^{n/2})\}$, com*

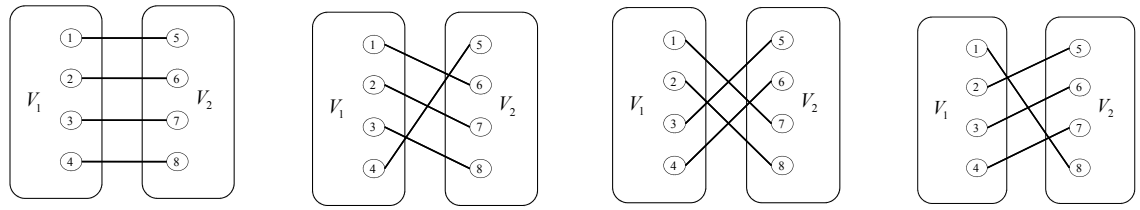
$$E^j = \{(a_i, b_{f_3(i,j)}) : i = 1\ldots n/2\}, \forall j \in \{1,\ldots,n/2\}, \tag{3.2}$$

onde,

$$f_3(i,j) = \begin{cases} i+j-1 & \text{Se } i+j-1 \leq n/2 \\ i+j-1-n/2 & \text{caso contrário.} \end{cases}$$

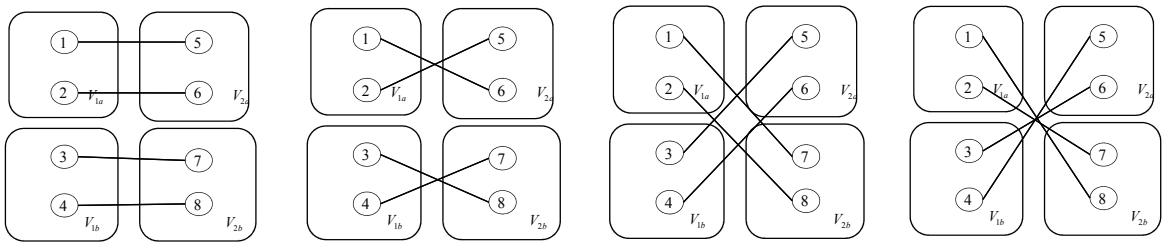
A Figura 3.6 mostra os fatores da 1-fatoração canônica para um grafo bipartido completo $K_{4,4}$.

O método de construção de 1-fatorações para $K_{n/2,n/2}$ pode também ser generalizado. Suponha inicialmente $n/2$ como sendo um número par. O método consiste em particionar o subconjunto de vértices V_1 em $\{V_{1a}, V_{1b}\}$ e V_2 em $\{V_{2a}, V_{2b}\}$, com $|V_{1a}| = |V_{1b}| = |V_{2a}| = |V_{2b}| = n/4$. Uma 1-fatoração é obtida para cada subgrafo bipartido completo de G do



(a) G^1 (b) G^2 (c) G^3 (d) G^4

Figura 3.6: 1-fatoração canônica para um grafo bipartido completo do tipo $(4,4)$.



(a) G^1 (b) G^2 (c) G^3 (d) G^4

Figura 3.7: 1-fatoração para um grafo bipartido completo com oito vértices, obtida por meio da aplicação método generalizado. A 1-fatoração canônica foi usada em cada um dos subgrafos.

tipo $(n/4, n/4)$, definido pelas partições $\{V_{1a}, V_{2a}\}$,$\{V_{1b}, V_{2b}\}$,$\{V_{1a}, V_{2b}\}$ e $\{V_{1b}, V_{2a}\}$. Para construir os primeiros $n/4$ fatores da 1-fatoração de G, é feita a união de algum fator da 1-fatoração associada à partição $\{V_{1a}, V_{2a}\}$ com algum fator da 1-fatoração associada à partição $\{V_{1b}, V_{2b}\}$. O processo é repetido usando pares de fatores não utilizados de 1-fatorações associadas às partições $\{V_{1a}, V_{2a}\}$ e $\{V_{1b}, V_{2b}\}$ até que $n/4$ fatores da 1-fatoração de G sejam obtidos. Os $n/4$ fatores restantes da 1-fatoração de G são obtidos por meio da combinação das 1-fatorações associadas às partições $\{V_{1a}, V_{2b}\}$ e $\{V_{1b}, V_{2a}\}$ de forma semelhante ao que foi feito para obter os $n/4$ primeiros fatores.

A Figura 3.7 ilustra a aplicação do método na construção de uma 1-fatoração para o grafo bipartido completo com oito vértices, com partição $\{V_1 = \{1,2,3,4\}, V_2 = \{5,6,7,8\}\}$. Os subconjuntos de vértices foram particionados em $V_{1a} = \{1,2\}$, $V_{1b} = \{3,4\}$, $V_{2a} = \{5,6\}$ e $V_{2b} = \{7,8\}$. A 1-fatoração canônica foi utilizada em cada subgrafo bipartido completo do tipo $(2,2)$, definidos por cada uma das partições $\{V_{1a}, V_{2a}\}$,$\{V_{1b}, V_{2b}\}$,$\{V_{1a}, V_{2b}\}$ e $\{V_{1b}, V_{2a}\}$.

O método pode ser estendido para quando $n/2$ for um número ímpar. Neste caso, o conjunto V_1 é particionado em $\{V_{1a}, V_{1b}\}$e V_2 em $\{V_{2a}, V_{2b}\}$, com $|V_{1a}| = \lceil n/4 \rceil$, $|V_{1b}| = \lfloor n/4 \rfloor$, $|V_{2a}| = \lfloor n/4 \rfloor$ e $|V_{2b}| = \lceil n/4 \rceil$. Vértices artificiais são adicionados aos elementos da partição com $\lfloor n/4 \rfloor$ vértices $V'_{2a} = V_{2a} \cup \{x\}$ e $V'_{1b} = V_{1b} \cup \{y\}$. Uma 1-fatoração é obtida para cada grafo bipartido completo do tipo $(\lceil n/4 \rceil, \lceil n/4 \rceil)$ definido pelas partições $\{V_{1a}, V'_{2a}\}$

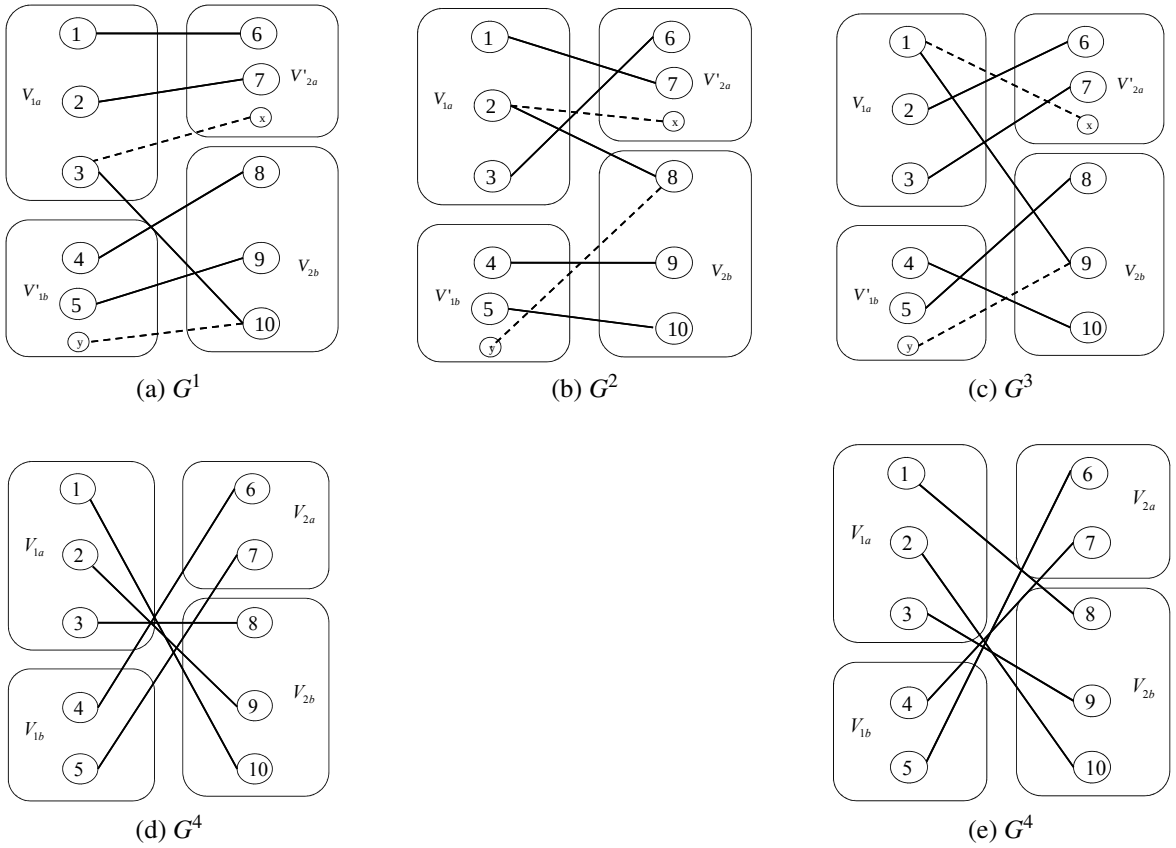
e $\{V'_{1b}, V_{2b}\}$. Para construir os primeiros $\lceil n/4 \rceil$ fatores da 1-fatoração de G, é feita uma combinação de algum fator da 1-fatoração associada à partição $\{V_{1a}, V'_{2a}\}$ com algum fator da 1-fatoração associada à partição $\{V'_{1b}, V_{2b}\}$. Essa combinação de fatores considera todas as arestas não incidentes aos vértices artificiais e também considera uma nova aresta (v_1, v_2) formada pelo vértice v_1 incidente a x no 1-fator associado à primeira partição e v_2 incidente a y no 1-fator associado à segunda partição. O processo é repetido usando pares de fatores não utilizados de 1-fatorações associadas às partições $\{V_{1a}, V'_{2a}\}$ e $\{V'_{1b}, V_{2b}\}$ até que $\lceil n/4 \rceil$ fatores da 1-fatoração de G sejam obtidos. Para obter os $\lfloor n/4 \rfloor$ fatores restantes da 1-fatoração de G, deve-se obter uma 1-fatoração para o grafo bipartido completo do tipo $(\lceil n/4 \rceil, \lceil n/4 \rceil)$ definido pela partição $\{V_{1a}, V_{2b}\}$, e uma 1-fatoração para o grafo bipartido completo do tipo $(\lfloor n/4 \rfloor, \lfloor n/4 \rfloor)$ definido pela partição $\{V_{1b}, V_{2a}\}$. A primeira 1-fatoração deve possuir um fator que inclua todas as arestas com uma extremidade em V_{1a} e outra extremidade em V_{2b} já presentes nos primeiros 1-fatores da 1-fatoração de G. Caso contrário, pode-se modificar a 1-fatoração obtida por meio de uma bijeção sobre o conjunto de vértices para que ela possua o 1-fator predeterminado. Esse 1-fator é ignorado. Os $\lfloor n/4 \rfloor$ fatores restantes da 1-fatoração de G são obtidos por meio da combinação dos fatores de cada uma das duas partições, de forma semelhante ao que foi feito na obtenção dos primeiros $\lceil n/4 \rceil$ fatores.

A Figura 3.8 ilustra a 1-fatoração obtida para o grafo $K_{5,5}$ por meio da aplicação do método generalizado. As Figuras (a)-(c) exibem os três primeiros fatores que foram obtidos através da 1-fatoração canônica aplicada aos grafos bipartidos completos referentes às partições $\{V_{1a}, V'_{2a}\}$ e $\{V'_{1b}, V_{2b}\}$. As arestas tracejadas, incidentes aos vértices artificiais, não fazem parte dos 1-fatores de G. As Figuras (d) e (e) correspondem aos fatores obtidos por meio de 1-fatorações para os grafos bipartidos completos associados às partições $\{V_{2b}, V_{1a}\}$ e $\{V_{1a}, V_{2b}\}$. A 1-fatoração canônica referente à partição $\{V_{2b}, V_{1a}\}$ precisou ser modificada para que contivesse o fator $f^* = \{(3,10),(2,8),(1,9)\}$ de arestas já presentes nos três primeiros fatores de G.

3.3.3 Procedimento construtivo

O espaço de busca para o PTVEP, considerando as vizinhanças definidas na seção 3.2, pode ser desconectado para certos valores de n. Isso significa que, partindo-se de uma solução qualquer, talvez não seja possível alcançar uma determinada solução, aplicando-se apenas os movimentos definidos pelas vizinhanças descritas. Nesse contexto, deseja-se obter a maior quantidade de 1-fatorações possivelmente desconectadas de K_n, aplicando-se o método generalizado descrito.

Uma 1-fatoração para um grafo bipartido completo G do tipo $(n/2, n/2)$, com $n/2$ par, pode ser construída a partir da combinação de quatro 1-fatorações F^1, F^2, F^3 e F^4 para um grafo bipartido completo do tipo $(n/4, n/4)$. Nota-se, pela simetria do método, que tro-



(a) G^1 (b) G^2 (c) G^3

(d) G^4 (e) G^4

Figura 3.8: 1-fatoração para o grafo $K_{5,5}$ obtida por meio da aplicação do método generalizado. As arestas tracejadas (incidentes aos vértices artificiais) não fazem parte dos 1-fatores de G.

car a ordem de escolha das 1-fatorações F^1 e F^2 gera 1-fatorações conectadas. O mesmo acontece em relação à ordem de escolha das 1-fatorações F^3 e F^4. Trocar a ordem de escolha do primeiro par de 1-fatorações, F^1e F^2, com o segundo par, F^3 e F^4, também gera 1-fatorações conectadas. No caso em que $n/2$ é ímpar, uma 1-fatoração para o grafo G pode ser construída a partir de três 1-fatorações F^1, F^2, F^3 para um grafo bipartido completo do tipo $(\lceil n/4 \rceil, \lceil n/4 \rceil)$ e uma 1-fatoração F^4 para um grafo bipartido completo do tipo $(\lfloor n/4 \rfloor, \lfloor n/4 \rfloor)$. Da mesma forma, trocar a ordem de escolha entre F^1 e F^2 gera 1-fatorações conectadas. Uma 1-fatoração para K_n (n par) pode ser construída a partir de duas 1-fatorações F^1, F^2 para $K_{n/2+resto(n/2,2)}$ e uma 1-fatoração F^3 para um grafo bipartido completo do tipo $(n/2, n/2)$. Da mesma forma, trocar a ordem entre F^1e F^2 gera 1-fatorações conectadas. Tais 1-fatorações claramente conectadas são evitadas.

Seja $B^*(i)$, com i par, o conjunto de todas as 1-fatorações que podem ser construídas para um grafo bipartido completo do tipo $(i/2, i/2)$, aplicando-se o método generalizado. Denote por $b(i)$ a 1-fatoração canônica para o grafo bipartido completo do tipo $(i/2, i/2)$. Denote por $combina(F^1,\ F^2,\ F^3,\ F^4)$ uma 1-fatoração para o grafo bipartido completo obtida pela

combinação das 1-fatorações F^1, F^2, F^3 e F^4. $B^*(i)$ pode ser definido pela seguinte fórmula de indução:

1. $b(i) \in B^*(i)$, para $i = 2, 4, ...$
2. Se $i/2$ é par e $F^1, F^2, F^3, F^4 \in B^*(i/2)$, então $combina(F^1, F^2, F^3, F^4) \in B^*(i)$.
3. Se $i/2$ é ímpar, $F^1, F^2, F^3 \in B^*(i/2+1)$ e $F^4 \in B^*(i/2-1)$, então $combina(F^1, F^2, F^3, F^4) \in B^*(i)$.

Um conjunto $B(i)$ é construído a partir de $B^*(i)$, eliminando aquelas 1-fatorações citadas que são claramente conectadas. Além disso, as definições 2 e 3 são aplicadas apenas para $i \geq 8$, pois os espaços de 1-fatorações para os grafos $K_{1,1}$, $K_{2,2}$ e $K_{3,3}$ são claramente conectados (possuem apenas a 1-fatoração canônica). Uma vez definida a ordem das 1-fatorações F^1, F^2, F^3, F^4, a ordem de escolha dos 1-fatores podem gerar 1-fatorações diferentes, mas todas elas são conectadas pela vizinhança $N4$. Portanto, é suficiente escolher apenas uma forma de combiná-las.

O algoritmo de programação dinâmica 2 constrói os conjuntos $B(i)$, para $i = 2, ..., n$, para um dado valor de n (par). Cada conjunto $B(i)$ é representado por uma lista de 1-fatorações indexadas por números inteiros. A notação $B(i)[j]$ denota a j-ésima 1-fatoração da lista $B(i)$. Inicialmente, cada lista $B(i)$ é inicializada com uma única 1-fatoração (canônica) para o grafo bipartido completo do tipo $(i/2, i/2)$ (linha 1). Em seguida, as definições (1) ou (2), dependendo se $i/2$ é par ou ímpar, são aplicadas para i de 8 até n, nesta ordem, para cada escolha possível das quatro 1-fatorações $F^1, F^2, F^3, F^4 \in B(i/2)$, quando $i/2$ é par, (linhas 4-14) ou $F^1, F^2, F^3 \in B(i/2+1)$ e $F^4 \in B(i/2-1)$, quando $i/2$ é ímpar (linhas 16-24). Os índices j_2 e j_4 são inicializados com j_1 e j_3, respectivamente, nas linhas 5, 7 e 17 para evitar 1-fatorações isomorfas que podem ser obtidas trocando-se a ordem entre as 1-fatorações F^1 e F^2 ou F^3 e F^4. O teste da linha 8 evita a geração de 1-fatorações isomorfas obtidas trocando-se a ordem de escolha do primeiro par de 1-fatorações, F^1e F^2, com o segundo par de 1-fatorações, F^3 e F^4.

Seja $C^*(i)$ (i par) o conjunto de todas as 1-fatorações que podem ser construídas para K_i, aplicando-se o método generalizado. Denote por $c(i)$ a 1-fatoração canônica para K_i. Denote por $combina(F^1, F^2, F^3)$ a 1-fatoração para o grafo completo obtida pela combinação das 1-fatorações F^1, F^2 e F^3. $C^*(i)$ pode ser definido pela seguinte fórmula de indução:

1. $c(i) \in C^*(i)$, para $i = 2, 4, ...$
2. Se $i/2$ é par, $F^1, F^2 \in C^*(i/2)$ e $F^3 \in B(i)$ então $combina(F^1, F^2, F^3) \in C^*(i)$.
3. Se $i/2$ é ímpar, $F^1, F^2 \in C^*(i/2+1)$ e $F^3 \in B(i)$ então $combina(F^1, F^2, F^3) \in C^*(i)$.

Algoritmo 2: Construção de B(i)

```
1  inicializa(B(i), b(i)), para i = 2,4,8,...,n;
2  para i = 8,10,...,n faça
3      se i/2 é par então
4          para j1 = 1,...,tamanho(B(i/2)) faça
5              para j2 = j1,...,tamanho(B(i/2)) faça
6                  para j3 = 1,...,tamanho(B(i/2)) faça
7                      para j4 = j3,...,tamanho(B(i/2)) faça
8                          se  j3 > j1 ou ( j3 = j1 e j4 ≥ j2) então
9                              insere(B(i), combina(B(i/2)[j1] B(i/2)[j2], B(i/2)[j3],
                               B(i/2)[j4]) )
10                         fim
11                     fim
12                 fim
13             fim
14         fim
15     senão
16         para j1 = 1,...,tamanho(B(i/2+1)) faça
17             para j2 = j2,...,tamanho(B(i/2+1)) faça
18                 para j3 = 1,...,tamanho(B(i/2+1)) faça
19                     para j4 = 1,...,tamanho(B(i/2−1)) faça
20                         insere(B(i), combina(B(i/2+1)[j1], B(i/2+1)[j2], B(i/2+1)[j3],
                           B(i/2-1)[j4]) )
21                     fim
22                 fim
23             fim
24         fim
25     fim
26 fim
```

Um conjunto $C(i)$ é construído a partir de $C^*(i)$, eliminando aquelas 1-fatorações citadas que são claramente conectadas. Além disso, as definições 2 e 3 são aplicadas apenas para $i \geq 8$, pois os espaços de 1-fatorações para os grafos K_2, K_4 e K_6 são claramente conectados (possuem apenas uma 1-fatoração, a canônica). Uma vez definida a ordem das 1-fatorações F^1, F^2 e F^3, a ordem de escolha dos 1-fatores podem gerar 1-fatorações diferentes, mas todas elas conectadas pela vizinhança $N4$. Por isso, é suficiente escolher apenas uma forma de combiná-las.

O algoritmo de programação dinâmica 3 constrói os conjuntos $C(i)$, para $i = 1,...,n$. Cada conjunto $C(i)$ é representado por uma lista de 1-fatorações indexadas por números inteiros. A notação $C(i)[j]$ denota a j-ésima 1-fatoração da lista $C(i)$. A Tabela 3.1 mostra a quantidade de 1-fatorações obtidas pelos algoritmos, para i de 8 até 20. O conjunto $B(12)$, por exemplo, contém duas 1-fatorações: $b(12)$ e $combina(b(6),b(6),b(6),b(6))$. O con-

Algoritmo 3: Construção de C(i)

```
1  inicializa(C(i), c(i)), para i = 2,4,8,...,n;
2  para i = 8,10,...,n faça
3  |  m = i/2 + resto(i/2,2) ;
4  |  para j1 = 1,...,tamanho(C(m)) faça
5  |  |  para j2 = j1,...,tamanho(C(m)) faça
6  |  |  |  para j3 = 1,...,tamanho(B(i)) faça
7  |  |  |  |  insere(C(i), combina(C(m)[j1], C(m)[j2], B(i)[j3]) )
8  |  |  |  fim
9  |  |  fim
10 |  fim
11 fim
```

n	2	4	6	8	10	12	14	16	18	20
$\|B(n)\|$	1	1	1	2	2	2	7	7	13	7
$\|C(n)\|$	1	1	1	3	3	3	43	43	79	43

Tabela 3.1: Quantidade de 1-fatorações iniciais geradas pelos algoritmos de programação dinâmica.

junto $C(12)$, por exemplo, contém três 1-fatorações: $c(12)$, $combina(c(6), c(6), b(12))$ e $combina(c(6), c(6), combina(b(6), b(6), b(6), b(6)))$. Não existe uma garantia de que as soluções construídas sejam de fato desconectadas.

O conjunto $C(n)$ será então utilizado na geração de soluções iniciais para o PTVEP com n equipes. Escolhida uma 1-fatoração, uma ordem aleatória é determinada para os 1-fatores. Em seguida, as equipes reais são atribuídas aleatoriamente aos vértices do grafo K_n. As soluções construídas podem não ser viáveis, devido à restrição de que uma equipe não pode jogar mais do que três partidas em rodadas consecutivas em casa nem mais do que três partidas em rodadas consecutivas fora de casa.

3.4 Busca Local

Dois procedimentos de busca local serão considerados. Eles se diferem apenas na vizinhança utilizada no procedimento de busca. A primeira busca local considera a vizinhança $N1 \cup N2$. A segunda busca local considera a vizinhança $N3 \cup N4$.

Seja $v(X)$ a quantidade de restrições violadas, ou seja, o número de sequências diferentes de exatamente quatro partidas consecutivas realizadas todas sob a mesma condição (em casa ou fora de casa), por uma mesma equipe, na solução corrente X. Seja $d(X)$ a distância total viajada na solução corrente X. O custo de todos os possíveis movimentos na vizinhança considerada são calculados em cada iteração da busca local. São computados o número

$v(X')$ de restrições violadas e a distância total viajada $d(X')$ para cada vizinho X' da solução corrente X. Se existe pelo menos uma solução vizinha X', tal que $v(X') \leq v(X)$ e $d(X') < d(X)$, então a solução corrente X é substituída pela solução vizinha com menor distância percorrida, dentre todas satisfazendo a condição anterior. Caso contrário, se não existe um movimento capaz de melhorar a distância percorrida sem piorar a quantidade de violações, a solução corrente é substituída pelo vizinho que mais decresce o número de violações. Se não existe tal movimento que diminua o número de violações, então o procedimento de busca local termina, e a solução corrente (mínimo local) é retornada.

O custo de se avaliar $d(X')$ e $v(X')$ para cada vizinho X' da solução corrente X é $O(n)$ para os dois procedimentos de busca local.

3.5 Perturbação e Critério de Aceitação

A perturbação utilizada consiste em fazer aleatoriamente dois, três ou quatro movimentos aleatórios em $N3 \cup N4$. Seja $v(X)$ a quantidade de violações na solução corrente X e $v(X')$ a quantidade de violações na solução candidata X'. Seja $d(X)$ a distância total viajada na solução corrente X e $d(X')$ a distância total viajada na solução candidata X'. O critério de aceitação consiste em trocar a solução corrente X pela solução candidata X' se, e somente se,

1. $v(X') < v(X)$ (diminui a quantidade de violações) , ou
2. $v(X') = v(X)$ e $d(X') < d(X)$ (diminui a distância total viajada sem piorar a quantidade de violações) , ou
3. a solução corrente não é alterada há mais de 100 iterações e $v(X') \leq v(X)$ e $d(X') \leq 1,01\, d(X)$ (a distância total viajada não é 1% pior do que a solução corrente, e a quantidade de violações não piora).

O algoritmo mantém sempre uma cópia da melhor solução encontrada durante sua execução. Quando o critério de parada é satisfeito, a melhor solução é retornada, e o algoritmo termina.

3.6 Experimentos Computacionais

Esta seção descreve os experimentos computacionais realizados para avaliar a heurística proposta. A heurística foi codificada em *C++* e compilada com a versão *4.1.2* do *gcc* com a opção de otimização de código *-O3*. Os experimentos foram realizados em um computador com processador de 3,00 GHz, 2 Gbytes de memória principal e sistema operacional *Debian GNU/Linux 4.0*.

3.6.1 Instâncias

As instâncias utilizadas nos testes foram as mesmas propostas e utilizadas por [Melo, 2007]. São um total de 40 instâncias, sendo 20 delas com 18 equipes e outras 20 com 20 equipes. As distâncias são as mesmas das instâncias circ18 e circ20 do Problema do Torneio com Viagens, ambas disponíveis em [Trick, 2009]. Dentre as instâncias de mesmo tamanho, 10 são balanceadas (i.e., cada equipe joga pelo menos $n/2-1$ jogos em casa e pelo menos $n/2-1$ jogos fora) e as outras 10 são não balanceadas. [Melo, 2007] mostrou que duas instâncias de cada tamanho são inviáveis.

3.6.2 Busca local

Um algoritmo de múltiplos inícios com busca local consiste em dois passos: (i) criar uma solução inicial e (ii) efetuar uma busca local para melhorar a qualidade da solução. Os passos (i) e (ii) são repetidos até que uma condição de parada seja satisfeita.

O primeiro experimento avalia o uso das vizinhanças $N_1 \cup N_2$ e $N_3 \cup N_4$ no procedimento de busca local. Um algoritmo de múltiplos inícios foi executado por um minuto usando cada uma das vizinhanças. A solução inicial foi construída aleatoriamente dentre todas as 1-fatorações construídas na seção 3.3.3.

A Tabela 3.2 mostra os resultados computacionais. A primeira coluna mostra o nome da instância. A segunda e a terceira coluna mostram o valor médio e o valor da melhor solução obtida usando a vizinhança $N_1 \cup N_2$. A quarta e a quinta coluna mostram o valor médio e o valor da melhor solução obtida usando a vizinhança $N_3 \cup N_4$. A última coluna mostra a melhoria (i.e. redução) percentual do valor da melhor solução encontrada usando a vizinhança $N_3 \cup N_4$ comparado com o valor da melhor solução encontrada usando a vizinhança $N_1 \cup N_2$.

Como era esperado, a busca local usando a vizinhança $N_3 \cup N_4$ funcionou melhor do que o algoritmo usando a vizinhança $N_1 \cup N_2$. A busca local em $N_3 \cup N_4$ foi capaz de encontrar soluções viáveis para todas as instâncias consideradas, enquanto a busca local em $N_1 \cup N_2$ não encontrou nenhuma solução viável para seis instâncias. Para as instâncias em que ambas encontraram soluções viáveis, a busca local em $N_3 \cup N_4$ encontrou soluções com valores pelo menos 10,91% melhores do que as soluções encontradas usando a vizinhança $N_1 \cup N_2$.

A busca local em $N_3 \cup N_4$ foi então selecionada para ser utilizada na heurística ILS.

3.6.3 *Heurística ILS*

O segundo experimento compara o desempenho da heurística ILS proposta usando a 1-fatoração canônica e 1-fatoração binária na construção da solução inicial. As Tabelas 3.3 e 3.4 mostram os resultados numéricos obtidos após cinco minutos de execução da heurís-

	$N_1 \cup N_2$		$N_3 \cup N_4$		Melhoria (%)
instância	Média	Menor	Média	Menor	
circ18abal	1073	1010	945,5	884	12,48
circ18bbal	1065	1004	946,5	880	12,35
circ18cbal	1070	1006	941,8	878	12,72
circ18dbal	1068	994	940,6	872	12,27
circ18ebal	1067	990	942,3	882	10,91
circ18fbal	1068	1002	945,7	874	12,77
circ18gbal	1075	1028	942	866	15,76
circ18hbal	1068	996	940,9	882	11,45
circ18ibal	1072	1012	944,3	882	12,85
circ18jbal	1064	1018	938,2	878	13,75
circ18anonbal	1097	1044	964,1	876	16,09
circ18dnonbal	1060	1060	952,8	902	14,91
circ18enonbal	1098	1098	969,8	924	15,85
circ18fnonbal			974,1	898	
circ18gnonbal			961,1	900	
circ18hnonbal	1097	1066	973,2	912	14,45
circ18inonbal			968,9	914	
circ18jnonbal	1078	1038	949,3	886	14,64
circ20abal	1481	1416	1295	1226	13,42
circ20bbal	1481	1394	1292	1210	13,20
circ20cbal	1466	1422	1287	1212	14,77
circ20dbal	1477	1408	1293	1212	13,92
circ20ebal	1484	1396	1289	1200	14,04
circ20fbal	1470	1386	1285	1210	12,70
circ20gbal	1465	1402	1289	1214	13,41
circ20hbal	1480	1392	1300	1220	12,36
circ20ibal	1461	1376	1292	1198	12,94
circ20jbal	1481	1420	1290	1186	16,48
circ20anonbal			1315	1270	
circ20bnonbal	1524	1496	1314	1230	17,78
circ20cnonbal			1311	1232	
circ20dnonbal			1323	1250	
circ20enonbal	1552	1552	1337	1266	18,43
circ20gnonbal	1521	1486	1333	1250	15,88
circ20inonbal	1491	1458	1304	1216	16,60
circ20jnonbal	1446	1426	1297	1222	14,31
Média	1280,00	1226,53	1127,42	1055,94	14,12

Tabela 3.2: Resultados numéricos comparativos entre o uso das vizinhanças $N_1 \cup N_2$ e $N_3 \cup N_4$ no procedimento de busca local.

tica ILS proposta usando cada 1-fatoração na construção da solução inicial, para instâncias com 18 e 20 equipes, respectivamente.

Instância	Canônica	Binária	Melhoria(%)
circ18abal	844	824	2,37
circ18bbal	834	814	2,40
circ18cbal	802	816	-1,75
circ18dbal	828	818	1,21
circ18ebal	820	826	-0,73
circ18fbal	804	834	-3,73
circ18gbal	828	804	2,90
circ18hbal	832	830	0,24
circ18ibal	814	830	-1,97
circ18jbal	804	812	-1,00
circ18anonbal	832	826	0,72
circ18dnonbal	842	816	3,09
circ18enonbal	844	832	1,42
circ18fnonbal	838	834	0,48
circ18gnonbal	842	834	0,95
circ18hnonbal	872	844	3,21
circ18inonbal	860	866	-0,70
circ18jnonbal	828	832	-0,48
Média	831,56	827,33	0,51

Tabela 3.3: Resultados numéricos para a heurística ILS usando 1-fatoração canônica e binária como solução inicial (18 equipes).

A primeira coluna mostra o nome da instância. A segunda e a terceira coluna mostram as distâncias percorridas obtidas utilizando cada 1-fatoração. A última coluna mostra a melhoria (i.e. redução) em percentual no valor da solução obtida usando a 1-fatoração binária comparada com o valor obtido usando a 1-fatoração canônica.

Para as instâncias com 18 equipes, as soluções obtidas usando ambas 1-fatorações na construção da solução inicial foram bem parecidas. Para instâncias com 20 equipes, o uso da 1-fatoração binária melhorou a solução em pelo menos 16,67%, com uma melhoria média de 18,62%. Isso é devido ao fato de que para 20 equipes e 1-fatoração canônica, as vizinhanças N_3 e N_4 são equivalentes às vizinhanças N_1 e N_2, respectivamente. Nesse caso, a heurística fica presa na estrutura da 1-fatoração canônica e explora apenas uma pequena porção do espaço de soluções. Esse fenômeno não acontece quando a 1-fatoração binária é utilizada.

No próximo experimento, a heurística foi executada por um total de duas horas (o tempo dado para as heurísticas em [Melo, 2007]) para obter soluções melhores para cada instância. Uma vez que existe uma grande variedade de 1-fatorações disponíveis para construção da solução inicial, foi proposto um método para escolher uma 1-fatoração inicial que gere uma solução inicial promissora. Esse método se inicia com 79 1-fatorações não isomorfas para

Instância	Canônica	Binária	Melhoria(%)
circ20abal	1378	1148	16,69
circ20bbal	1400	1140	18,57
circ20cbal	1390	1134	18,42
circ20dbal	1380	1150	16,67
circ20ebal	1370	1136	17,08
circ20fbal	1380	1142	17,25
circ20gbal	1372	1124	18,08
circ20hbal	1382	1134	17,95
circ20ibal	1374	1138	17,18
circ20jbal	1376	1128	18,02
circ20anonbal	1502	1174	21,84
circ20bnonbal	1420	1142	19,58
circ20cnonbal	1438	1140	20,72
circ20dnonbal	1422	1166	18,00
circ20enonbal	1452	1170	19,42
circ20gnonbal	1470	1150	21,77
circ20inonbal	1408	1144	18,75
circ20jnonbal	1382	1118	19,10
Média	1405,33	1143,22	18,62

Tabela 3.4: Resultados numéricos para a heurística ILS usando 1-fatoração canônica e binária como solução inicial (20 equipes).

K_{18} e 43 para K_{20} (veja tabela 3.1). Ele possui dois parâmetros: o número s de estágios e o tempo t gasto em cada estágio. No primeiro estágio, cada 1-fatoração é utilizada para construir um solução inicial e a heurística ILS é executada para cada solução inicial totalizando um tempo t de execução. As $z/\sqrt[s]{f}$ melhores soluções encontradas no estágio corrente passam para o estágio seguinte, onde z é o número de soluções no estágio corrente e f é o número de 1-fatorações construídas inicialmente. A fração de soluções que passam para o próximo estágio é calculada de tal forma que apenas uma solução chega ao estágio final. O processo é repetido até que s estágios sejam completados e a melhor solução é retornada. O algoritmo 4 descreve o método. Na linha 1, o método generalizado é utilizado para construir o conjunto F de 1-fatorações. Cada uma das soluções iniciais consideradas em um dado estágio $j = 1, \ldots, s$ são investigadas nas linhas de 6 a 8 e trocadas pela solução obtida pela aplicação da heurística ILS por $t/|F|$ unidades de tempo. Nas linhas 10 e 11, as melhores soluções são selecionadas para passar para o próximo estágio, enquanto as demais soluções são descartadas.

Neste experimento, a heurística 2 que usa a estratégia descrita para escolha da solução inicial é comparada com a heurística que seleciona aleatoriamente uma 1-fatoração como solução inicial. A heurística 2 gasta um total de três minutos escolhendo a solução inicial, distribuída em três estágios. Em seguida, a heurística ILS é executada para a solução inicial

Algoritmo 4: Método para escolha da solução inicial.

Use o método generalizado para obter um conjunto completo F de 1-fatorações;
$f \leftarrow |F|$;
para $j = 1, \ldots, s$ **faça**
    $F' \leftarrow \emptyset$;
    para $e \in F$ **faça**
        Escolha a solução e e a remova de F: $F \leftarrow F - \{e\}$;
        Obtenha uma nova solução e' aplicando a heurística ILS a partir de e por $t/|F|$ unidades de tempo;
        Considere a solução e' como candidata para o próximo estágio: $F' \leftarrow F' \cup \{e'\}$;
    fim
    Mantenha em F' as $\lceil |F|/\sqrt[s]{f} \rceil$ melhores soluções e descarte as demais;
    $F \leftarrow F'$;
fim
Retorna a melhor 1-fatoração em F ;

selecionada por 117 minutos. A heurística aleatória escolhe uma 1-fatoração qualquer, com exceção da 1-fatoração canônica para $n = 20$, para construção da solução inicial. Em seguida, a heurística ILS é executada por 120 minutos.

As Tabelas 3.5 e 3.6 mostram os resultados numéricos obtidos por cada algoritmo. A primeira coluna mostra o nome da instância. A segunda e a terceira coluna mostram os valores numéricos das soluções obtidas por cada algoritmo. A última coluna mostra a melhoria (i.e. a redução) em porcentagem no valor da solução obtida pelo algoritmo 2 comparado com o valor obtido pelo algoritmo aleatório.

Os resultados de ambas as heurísticas foram parecidos para as instâncias com 18 equipes. Para instâncias com 20 equipes, a heurística 2 obteve resultados melhores para 61% das instâncias viáveis e melhorou o resultado médio em 0,39%.

No último experimento, uma rápida execução da heurística proposta com tempo limite de um segundo foi comparado com os melhores resultados obtidos pelas heurísticas baseadas em programação linear inteira em [Melo, 2007], usando um computador similar ao usado neste trabalho. A 1-fatoração inicial foi escolhida aleatoriamente, uma vez que o tempo total de execução fixado foi muito curto.

A Tabela 3.7 mostra os resultados numéricos obtidos. A primeira coluna mostra o nome das instâncias. A segunda coluna mostra o valor da melhor solução obtida, dentre aquelas encontradas pelos quatro algoritmos descritos por [Melo, 2007], depois de duas horas de execução para cada algoritmo. A terceira coluna mostra o valor da solução obtida pela heurística de busca local iterada após um segundo de execução. A última coluna mostra a melhoria (i.e., redução) percentual do valor da solução encontrada pela heurística ILS comparada com os melhores resultados apresentados em [Melo, 2007].

Instâncias	Aleatório	Heurística 2	Melhoria
circ18abal	776	790	-1,80
circ18bbal	796	810	-1,76
circ18cbal	794	802	-1,01
circ18dbal	794	788	0,76
circ18ebal	784	790	-0,77
circ18fbal	798	792	0,75
circ18gbal	784	792	-1,02
circ18hbal	792	780	1,52
circ18ibal	778	806	-3,60
circ18jbal	794	780	1,76
circ18anonbal	814	806	0,98
circ18dnonbal	800	810	-1,25
circ18enonbal	804	814	-1,24
circ18fnonbal	826	802	2,91
circ18gnonbal	802	782	2,49
circ18hnonbal	822	804	2,19
circ18inonbal	818	818	0,00
circ18jnonbal	790	782	1,01
Média	798,11	797,11	0,11

Tabela 3.5: Resultados numéricos comparativos entre a heurística 2 e a heurística aleatória (18 equipes).

A heurística proposta supera claramente as heurísticas descritas em [Melo, 2007]. Os resultados mostraram que com apenas um segundo de execução a heurística ILS melhora a melhor solução conhecida anteriormente na literatura em pelo menos 11,42% para cada instância.

Instância	Aleatório	Heurística 2	Melhoria
circ20abal	1098	1094	0,36
circ20bbal	1082	1090	-0,74
circ20cbal	1072	1082	-0,93
circ20dbal	1104	1100	0,36
circ20ebal	1100	1076	2,18
circ20fbal	1076	1072	0,37
circ20gbal	1082	1068	1,29
circ20hbal	1094	1094	0,00
circ20ibal	1078	1084	-0,56
circ20jbal	1096	1086	0,91
circ20anonbal	1106	1130	-2,17
circ20bnonbal	1096	1082	1,28
circ20cnonbal	1104	1096	0,72
circ20dnonbal	1136	1140	-0,35
circ20enonbal	1100	1128	-2,55
circ20gnonbal	1130	1078	4,60
circ20inonbal	1092	1082	0,92
circ20jnonbal	1084	1070	1,29
Média	1096,11	1091,78	0,39

Tabela 3.6: Resultados numéricos comparativos entre a heurística 2 e a heurística aleatória (20 equipes).

Instância	Melhor	ILS(1s)	Melhoria(%)
circ18abal	1106	912	17,54
circ18bbal	1100	896	18,55
circ18cbal	1038	892	14,07
circ18dbal	1096	882	19,53
circ18ebal	1074	892	16,95
circ18fbal	1060	910	14,15
circ18gbal	1100	894	18,73
circ18hbal	1094	880	19,56
circ18ibal	1102	894	18,87
circ18jbal	1078	878	18,55
circ18anonbal	1124	940	16,37
circ18dnonbal	1060	914	13,77
circ18enonbal	1092	922	15,57
circ18fnonbal	1098	956	12,93
circ18gnonbal	1098	924	15,85
circ18hnonbal	1110	944	14,95
circ18inonbal	1104	932	15,58
circ18jnonbal	1102	898	18,51
circ20abal	1520	1236	18,68
circ20bbal	1530	1252	18,17
circ20cbal	1470	1234	16,05
circ20dbal	1464	1238	15,44
circ20ebal	1526	1214	20,45
circ20fbal	1546	1236	20,05
circ20gbal	1536	1210	21,22
circ20hbal	1516	1268	16,36
circ20ibal	1544	1238	19,82
circ20jbal	1484	1222	17,65
circ20anonbal	1502	1270	15,45
circ20bnonbal	1522	1258	17,35
circ20cnonbal	1488	1318	11,42
circ20dnonbal	1510	1294	14,30
circ20enonbal	1574	1250	20,58
circ20gnonbal	1540	1278	17,01
circ20inonbal	1516	1236	18,47
circ20jnonbal	1516	1220	19,53
Média	1303,89	1078,67	17,17

Tabela 3.7: Resultado comparativo entre ILS (um segundo de execução) e o melhor resultado conhecido anteriormente na literatura.

Capítulo 4

Limites Duais para o PTVEP

Um limite dual para um problema de minimização (resp. maximização) é um limite inferior (resp. superior) para a função objetivo. Este capítulo apresenta novos métodos para obtenção de limites duais para o PTVEP baseados nos modelos de programação linear inteira de [Melo, 2007] e [Easton et al., 2001].

Na seção 4.1, é apresentado um modelo de programação linear inteira para o PTVEP proposto por [Melo, 2007]. Na seção 4.2, são apresentados quatro novos modelos relaxados para o PTVEP, baseados em [Melo, 2007] e [Easton et al., 2001]. Na seção 4.3, são apresentados os resultados computacionais deste capítulo.

4.1 Formulação de Programação Linear Inteira

Melo [2007] propôs três modelos de programação linear inteira para o PTVEP. Segue a descrição do modelo que obteve os melhores limites duais através de sua relaxação linear. Suponha que sejam dados (i) Um conjunto com n (par) equipes $V = \{1,\ldots,n\}$, (ii) uma matriz $D(n \times n)$ de distâncias (não necessariamente simétrica), onde cada elemento $d_{ij} \geq 0$ de D contém a distância entre as cidades sedes das equipes i e j e (iii) um conjunto de atribuições casa-fora J. As variáveis de decisão da formulação são:

$$w^1_{tik} = \begin{cases} 1, & \text{se a equipe t viaja de sua cidade sede para a cidade da equipe i na rodada k} \\ & \text{e retorna para sua cidade na rodada k+1} \\ 0, & \text{caso contrário,} \end{cases}$$

$$w^2_{tijk} = \begin{cases} 1, & \text{se a equipe t parte de sua cidade sede e visita as cidades i e j nas} \\ & \text{rodadas k e k+1, respectivamente, e então,} \\ & \text{retorna para sua cidade sede na rodada k+2} \\ 0, & \text{caso contrário} \end{cases}$$

e

$$w^3_{tijmk} = \begin{cases} 1, & \text{se a equipe t parte de sua cidade sede e visita as cidades das equipes} \\ & \text{i,j,m nas rodadas k,k+1e k+2, respectivamente,} \\ & \text{e então, retorna para sua cidade sede na rodada k+3} \\ 0, & \text{caso contrário.} \end{cases}$$

As variáveis representam sequências de partidas consecutivas que cada equipe realiza fora de casa. Duas rodadas fictícias (indexadas por -1 e 0) são criadas para simplificar a formulação. As variáveis com índice $k \in \{-1,0\}$ são fixadas em zero. Os custos auxiliares $c_{ij} = d_{ij} + d_{ji}$, $c_{ijm} = d_{ij} + d_{jm} + d_{mi}$ e $c_{ijml} = d_{ij} + d_{jm} + d_{ml} + d_{li}$ representam os custos acumulados das sequências de partidas consecutivas que cada equipe i realiza fora de casa. Seja $r = n-1$ o índice da última rodada. As equações (4.1)-(4.9) definem um modelo de programação linear inteira para o PTVEP.

$$\min F(w^1, w^2, w^3) = \sum_{k=1}^{r} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ (j,i)\in J}}^{n} [c_{ij} w^1_{ijk} + \sum_{\substack{m=1 \\ (m,i)\in J \\ m\neq j}}^{n} (c_{ijm} w^2_{ijmk} + \sum_{\substack{l=1 \\ (l,i)\in J \\ l\neq j, l\neq m}}^{n} c_{ijml} w^3_{ijmlk})] \quad (4.1)$$

Sujeito a:

$$\sum_{i=1}^{n} \sum_{\substack{j=1 \\ j\neq i}}^{n} [\sum_{k\in\{-1,0\}} w^1_{ijk} + \sum_{\substack{m=1 \\ m\notin\{i,j\}}}^{n} (\sum_{k\in\{-1,0,r\}} w^2_{ijmk} + \sum_{\substack{l=1 \\ l\notin\{i,j,m\}}}^{n} \sum_{k\in\{-1,0,r-1,r\}} w^3_{ijmlk})] = 0 \quad (4.2)$$

$$\sum_{k=1}^{r} \{w^1_{ijk} + \sum_{\substack{m=1 \\ (m,i)\in J \\ m\notin\{i,j\}}}^{n} [(w^2_{ijmk} + w^2_{imjk}) + \sum_{\substack{l=1 \\ (l,i)\in J \\ l\notin\{i,j,m\}}}^{n} (w^3_{ijmlk} + w^3_{imjlk} + w^3_{imljk})]\} = 1,$$

$$\forall (j,i) \in J, \quad (4.3)$$

$$\sum_{\substack{j=1\\(j,i)\in J}}^{n}\{w^1_{ijk}+\sum_{\substack{m=1\\(m,i)\in J\\m\notin\{i,j\}}}^{n}[(w^2_{ijmk}+w^2_{ijm,k-1})+\sum_{\substack{l=1\\(l,i)\in J\\l\notin\{i,j,m\}}}^{n}(w^3_{ijmlk}+w^3_{ijml,k-1}+w^3_{ijml,k-2})]\}$$

$$+\sum_{\substack{j=1\\(i,j)\in J}}^{n}\{w^1_{jik}+\sum_{\substack{m=1\\(m,i)\in J\\m\notin\{i,j\}}}^{n}[(w^2_{jimk}+w^2_{jmi,k-1})+\sum_{\substack{l=1\\(l,i)\in J\\l\notin\{i,j,m\}}}^{n}(w^3_{jimlk}+w^3_{jmil,k-1}+w^3_{jmli,k-2})]\}=1,$$

$$i=1,\ldots,n,\ k=1,\ldots,r \tag{4.4}$$

$$\sum_{\substack{j=1\\(j,i)\in J}}^{n}\{w^1_{ijk}+w^1_{ij,k+1}+\sum_{\substack{m=1\\(m,i)\in J\\m\notin\{i,j\}}}^{n}[w^2_{ijm,k-1}+w^2_{ijmk}+w^2_{ijm,k+1}+$$

$$+\sum_{\substack{l=1\\(l,i)\in J\\l\notin\{i,j,m\}}}^{n}(w^3_{ijml,k-2}+w^3_{ijml,k-1}+w^3_{ijmlk}+w^3_{ijml,k+1})]\}\leq 1,$$

$$i=1,\ldots,n,\quad k=1,\ldots,r-1 \tag{4.5}$$

$$\sum_{q=k}^{k+3}\{\sum_{\substack{j=1\\(j,l)\in J}}^{n}[w^1_{ijq}+\sum_{\substack{m=1\\(m,i)\in J\\m\notin\{i,j\}}}^{n}((w^2_{ijmq}+w^2_{ijm,q-1})+\sum_{\substack{l=1\\(l,i)\in J\\l\notin\{i,j,m\}}}^{n}(w^3_{ijmlq}+w^3_{ijml,q-1}+w^3_{ijml,q-2}))]\}\geq 1,$$

$$i=1,\ldots,n,\quad k=1,\ldots,r-3 \tag{4.6}$$

$$w^1_{ijk}\in\{0,1\},i,j=1,\ldots,n,\ \text{com}\ i\neq j,\quad k=-1,\ldots,r \tag{4.7}$$

$$w^2_{ijmk}\in\{0,1\},i,j,m=1,\ldots,n,\ \text{com}\ |\{i,j,m\}|=3,\quad k=-1,\ldots,r \tag{4.8}$$

$$w^3_{ijmlk}\in\{0,1\},i,j,m,l=1,\ldots,n,\ \text{com}\ |\{i,j,m,l\}|=4,\quad k=-1,\ldots,r. \tag{4.9}$$

A função objetivo (4.1) minimiza a distância total percorrida. As restrições (4.2) fixam em zero as variáveis associadas às rodadas artificiais. As restrições (4.3) asseguram que cada jogo ocorra exatamente uma vez. As restrições (4.4) forçam que cada equipe i, em cada rodada, esteja disputando um jogo fora de casa, ou então, uma outra equipe esteja visitando a equipe i. As restrições (4.5) proíbem a equipe i de estar engajada em sequências incompatíveis de partidas consecutivas fora de casa. As restrições (4.6) garantem que cada equipe i jogue pelo menos uma partida fora de casa em um intervalo de quatro rodadas

consecutivas. As restrições (4.7)-(4.9) garantem a integralidade das variáveis. O modelo possui $O(n^5)$ variáveis e $O(n^2)$ restrições.

4.2 Limites Inferiores

Os melhores limites duais conhecidos anteriormente para o PTVEP foram obtidos através da relaxação linear do modelo apresentado na seção 4.1. Nesta seção, propõem-se quatro modelos de programação linear inteira que são relaxações combinatórias do modelo da seção 4.1.

4.2.1 Limite Inferior Independente

Uma ideia muito simples pode ser aplicada para obter um limite inferior para o PTVEP. Ela consiste em determinar, para cada equipe t, a mínima distância $dist(t)$ que cada equipe deve viajar para visitar todas as outras equipes que serão enfrentadas fora de casa, considerando apenas que cada equipe não pode visitar mais do que três equipes consecutivas sem que retorne à sua cidade sede. Esse problema é equivalente ao Problema de Roteamento de Veículos Capacitado [Dantzig e Ramser, 1959]. Embora seja *NP*-difícil, o problema pode ser resolvido rapidamente para instâncias de interesse neste trabalho. A soma dos limites $dist(t)$ para cada equipe t fornece um limite inferior *ILB* (do inglês, *Independent Lower Bound*) para o PTVEP [Easton et al., 2001].

Considere a relaxação do modelo de programação linear inteira da seção 4.1 formulada apenas pelas equações (4.1), (4.3) e (4.7)-(4.9), ignorando as variáveis artificiais. Como não existem restrições relacionando viagens entre duas equipes distintas, o problema pode ser decomposto por equipes. Para cada equipe $i \in V$, tem-se o seguinte subproblema :

$$\min F^i(w_i^1, w_i^2, w_i^3) \quad = \sum_{k=1}^{r} \sum_{\substack{j=1 \\ (j,i)\in J}}^{n} [c_{ij} w_{ijk}^1 + \sum_{\substack{m=1 \\ (m,i)\in J \\ m\neq j}}^{n} (c_{ijm} w_{ijmk}^2 + \sum_{\substack{l=1 \\ (l,i)\in J \\ l\neq j, l\neq m}}^{n} c_{ijml} w_{ijmlk}^3)] \quad (4.10)$$

Sujeito a:

$$\sum_{k=1}^{r} \{w_{ijk}^1 + \sum_{\substack{m=1 \\ (m,i)\in J \\ m\notin\{i,j\}}}^{n} [(w_{ijmk}^2 + w_{imjk}^2) + \sum_{\substack{l=1 \\ (l,i)\in J \\ l\notin\{i,j,m\}}}^{n} (w_{ijmlk}^3 + w_{imjlk}^3 + w_{imljk}^3)]\} = 1,$$

$$\forall j \in V \text{ tal que } (j,i) \in J, \quad (4.11)$$

$$w^1_{ijk} \in \{0,1\}, j = 1,\ldots,n, \text{ com } (j,i) \in J, \quad k = 1,\ldots,r \tag{4.12}$$

$$w^2_{ijmk} \in \{0,1\}, j,m = 1,\ldots,n, \text{ com } (j,i),(m,i) \in J, |\{i,j,m\}| = 3, \quad k = 1,\ldots,r \tag{4.13}$$

$$w^3_{ijmlk} \in \{0,1\}, j,m,l = 1,\ldots,n, \text{ com } (j,i),(m,i),(l,i) \in J, |\{i,j,m,l\}| = 4, \quad k = 1,\ldots,r. \tag{4.14}$$

A função objetivo 4.10 minimiza os custos relacionados às viagens da equipe i. As restrições (4.11) asseguram que i deve visitar cada adversário cuja partida deve ser realizada fora de casa. Pela natureza das variáveis, a equipe i não pode visitar mais do que três equipes consecutivas sem que retorne à sua cidade sede. Cada subproblema equivale ao Problema de Roteamento de Veículos Capacitado envolvido no cálculo do ILB. A soma $\sum_{i \in V} F^i$ do valor ótimo encontrado em cada subproblema fornece o ILB.

O modelo possui $O(n)$ restrições e $O(n^4)$ variáveis.

4.2.2 Modelo simplificado para obtenção do ILB

O cálculo do ILB não leva em consideração em qual rodada cada equipe i visita seus adversários. O modelo pode então ser simplificado eliminando os índices k, reduzindo, assim, a quantidade de variáveis do problema. As equações (4.15)-(4.19) formulam o modelo simplificado para o cálculo do ILB:

$$\min F_i(w^1_i, w^2_i, w^3_i) \quad = \sum_{\substack{j=1 \\ (j,i) \in J}}^{n} [c_{ij} w^1_{ij} + \sum_{\substack{m=1 \\ (m,i) \in J \\ m \neq j}}^{n} (c_{ijm} w^2_{ijm} + \sum_{\substack{l=1 \\ (l,i) \in J \\ l \neq j, l \neq m}}^{n} c_{ijml} w^3_{ijml})] \tag{4.15}$$

Sujeito a:

$$w^1_{ij} + \sum_{\substack{m=1 \\ (m,i) \in J \\ m \notin \{i,j\}}}^{n} [(w^2_{ijm} + w^2_{imj}) + \sum_{\substack{l=1 \\ (l,i) \in J \\ l \notin \{i,j,m\}}}^{n} (w^3_{ijml} + w^3_{imjl} + w^3_{imlj})] = 1, \tag{4.16}$$

$\forall j$ tal que $(j,i) \in J$,

$$w^1_{ij} \in \{0,1\}, j = 1,\ldots,n, \text{ com } (j,i) \in J \tag{4.17}$$

$$w^2_{ijm} \in \{0,1\}, j,m = 1,\ldots,n, \text{ com } |\{i,j,m\}| = 3, (j,i),(m,i) \in J \tag{4.18}$$

$$w^3_{ijml} \in \{0,1\}, j,m,l = 1,\ldots,n, \text{ com } |\{i,j,m,l\}| = 4, \quad (j,i),(m,i),(l,i) \in J. \tag{4.19}$$

O modelo possui $O(n)$ restrições e $O(n^3)$ variáveis.

4.2.3 ILB melhorado para instâncias não balanceadas

As restrições (4.4) são as que de fato dificultam a solução exata do problema. Elas relacionam viagens entre equipes diferentes, o que impossibilita a decomposição do modelo. Um novo modelo, considerando todas as restrições do modelo original da seção 4.1, exceto as restrições complicantes (4.4), será então considerado.

As restrições (4.5) adicionadas impõem que, para cada equipe, exista pelo menos uma partida em casa intercalando cada par de sequências de partidas consecutivas fora de casa, o que limita superiormente a quantidade de sequências de viagens do problema.

As restrições (4.6) adicionadas proíbem as equipes de realizarem mais do que três partidas consecutivas em casa, o que implica em um limite inferior na quantidade de sequências de viagens consecutivas de uma mesma equipe.

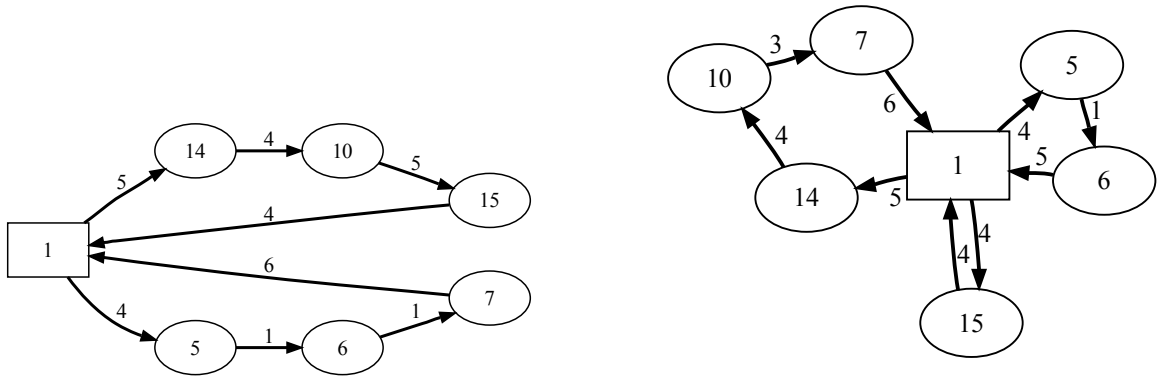
Com as restrições adicionadas, a quantidade de sequências de partidas consecutivas em casa não pode diferir de mais de uma unidade da quantidade de sequências de partidas consecutivas realizadas fora de casa, para cada equipe. Uma ideia semelhante foi aplicada em Urrutia et al. [2007] para limitar a quantidade de viagens para a versão espelhada do Problema do Torneio com Viagens.

A Figura 4.1 compara as soluções obtidas para os subproblemas referentes à equipe 1 no cálculo do ILB (Figura (a)) e do ILB melhorado (Figura (b)) para uma instância com 18 equipes. A solução para o ILB possui duas variáveis não nulas: $w^3_{1,14,10,15} = w^3_{1,5,6,7} = 1$. A distância percorrida pela equipe 1 na solução do ILB apresentada é 30. No cálculo do ILB melhorado, as restrições 4.5 e 4.6 são consideradas. Como a equipe i faz 11 partidas em casa, então a quantidade de sequências de partidas consecutivas fora de casa (i.e. número de variáveis não nulas) deve ser maior ou igual a três. Caso contrário, existiria pelo menos uma sequência de quatro partidas consecutivas em casa. A solução ótima apresentada possui três variáveis não nulas: $w^3_{1,14,10,7,4} = w^2_{1,5,6,10} = w^1_{1,15,15} = 1$. A distância percorrida pela equipe 1, nesse caso, é 36. A grande diferença entre a quantidade de partidas em casa e a quantidade de partidas fora de casa para a equipe 1, no exemplo apresentado, proporcionou uma melhora de 6 unidades de distância no valor do limite fornecido.

O modelo possui $O(n)$ restrições e $O(n^4)$ variáveis.

4.2.4 Modelo simplificado para obtenção do ILB melhorado

As restrições (4.5) e (4.6) do modelo anteriormente apresentado podem ser modeladas sem a necessidade de se conhecer em qual rodada cada adversário é visitado, o que possibilita uma simplificação do modelo.



(a) ILB. (b) ILB melhorado.

Figura 4.1: Comparação entre as soluções obtidas para os subproblemas referentes à equipe 1 no cálculo do ILB e do ILB melhorado para a instância circ18hnonbal.

Seja sc_i a quantidade de sequências de partidas consecutivas que a equipe i realiza em casa, e sf_i a quantidade de sequências de partidas consecutivas que a equipe i realiza fora de casa. Sejam c_i e f_i as quantidades de partidas que a equipe i realiza em casa e fora de casa, respectivamente. A quantidade de sequências de partidas consecutivas em casa não pode diferir em mais de uma unidade da quantidade de sequências de partidas consecutivas realizadas fora de casa, ou seja,

$$|sc_i - sf_i| \leq 1. \tag{4.20}$$

A quantidade de sequências de partidas consecutivas em casa deve ser no mínimo igual à $\lceil \frac{c_i}{3} \rceil$e no máximo igual à quantidade de partidas que i realiza em casa, ou seja,

$$\lceil \frac{c_i}{3} \rceil \leq sc_i \leq c_i. \tag{4.21}$$

Então,

$$\begin{aligned}
\lceil \frac{c_i}{3} \rceil &\leq sc_i \quad \text{, por (4.21)} \\
&\leq sf_i + 1 \quad \text{, por (4.20)} \\
\Rightarrow \lceil \frac{c_i}{3} \rceil - 1 \leq sf_i &= \sum_{\substack{j=1 \\ (j,i)\in J}}^{n} [w_{ij}^1 + \sum_{\substack{m=1 \\ (m,i)\in J \\ m\neq j}}^{n} (w_{ijm}^2 + \sum_{\substack{l=1 \\ (l,i)\in J \\ l\neq j, l\neq m}}^{n} w_{ijml}^3)]. \qquad (4.22)
\end{aligned}$$

o que limita inferiormente o valor de sf_i. Por outro lado,

$$\begin{aligned}
sf_i &\leq sc + 1 \quad \text{, por (4.20)} \qquad (4.23) \\
&\leq c_i + 1 \quad \text{, por (4.21)} \\
\Rightarrow sf_i &= \sum_{\substack{j=1 \\ (j,i)\in J}}^{n} [w_{ij}^1 + \sum_{\substack{m=1 \\ (m,i)\in J \\ m\neq j}}^{n} (w_{ijm}^2 + \sum_{\substack{l=1 \\ (l,i)\in J \\ l\neq j, l\neq m}}^{n} w_{ijml}^3)] \leq c_i + 1 \qquad (4.24)
\end{aligned}$$

o que limita superiormente o valor de sf_i.

Se uma solução não cumpre alguma das duas restrições (equações (4.22) e (4.24)), então não é possível associar rodadas às variáveis da solução de forma a obter uma solução que satisfaça as restrições 4.5 e 4.6. Se uma solução obedece as duas restrições, então é possível associar uma rodada k a cada variável de forma que a solução construída satisfaça as restrições 4.5 e 4.6 . Pode-se, então, afirmar que a substituição das restrições (4.5) e (4.6) pelas duas novas restrições (4.22) e (4.24) não influencia no valor do limite fornecido.

As equações (4.25)-(4.31) definem o modelo simplificado para obtenção do ILB melhorado:

$$\min F_i(w_i^1, w_i^2, w_i^3) = \sum_{\substack{j=1 \\ (j,i)\in J}}^{n} [c_{ij} w_{ij}^1 + \sum_{\substack{m=1 \\ (m,i)\in J \\ m\neq j}}^{n} (c_{ijm} w_{ijm}^2 + \sum_{\substack{l=1 \\ (l,i)\in J \\ l\neq j, l\neq m}}^{n} c_{ijml} w_{ijml}^3)] \qquad (4.25)$$

Sujeito a:

$$w_{ij}^1 + \sum_{\substack{m=1 \\ (m,i)\in J \\ m\notin\{i,j\}}}^{n} [(w_{ijm}^2 + w_{imj}^2) + \sum_{\substack{l=1 \\ (l,i)\in J \\ l\notin\{i,j,m\}}}^{n} (w_{ijml}^3 + w_{imjl}^3 + w_{imlj}^3)] = 1, \qquad (4.26)$$

$\forall j$ tal que $(j,i) \in J$,

$$\lceil \frac{c_i}{3} \rceil - 1 \leq \sum_{\substack{j=1 \\ (j,i)\in J}}^{n} [w^1_{ij} + \sum_{\substack{m=1 \\ (m,i)\in J \\ m\neq j}}^{n} (w^2_{ijm} + \sum_{\substack{l=1 \\ (l,i)\in J \\ l\neq j, l\neq m}}^{n} w^3_{ijml})] \leq c_i + 1 \tag{4.27}$$

(4.28)

$$w^1_{ij} \in \{0,1\}, j = 1,\ldots,n, \text{ com } (j,i) \in J \tag{4.29}$$

$$w^2_{ijm} \in \{0,1\}, j,m = 1,\ldots,n, \text{ com } |\{i,j,m\}| = 3, (j,i), (m,i) \in J \tag{4.30}$$

$$w^3_{ijml} \in \{0,1\}, j,m,l = 1,\ldots,n, \text{ com } |\{i,j,m,l\}| = 4, \quad (j,i), (m,i), (l,i) \in J. \tag{4.31}$$

O modelo possui $O(n)$ restrições e $O(n^3)$ variáveis.

4.3 Experimentos Computacionais

Esta seção descreve os experimentos computacionais realizados para avaliar os modelos propostos. Os programas foram codificados em C++ e compilados com a versão 4.1.2 do *gcc* com a opção de otimização de código -O3. Eles utilizam a biblioteca *API Concert Technology v.2.0* e o *software cplex v. 10.2* para descrever e solucionar os modelos de programação linear e programação linear inteira. Os experimentos foram realizados em um computador com processador de 3,00 GHz, 2 Gbytes de memória principal e sistema operacional *Debian GNU/Linux 4.0*. As instâncias de testes utilizadas foram as mesmas descritas na seção 3.6.1.

Os cinco modelos apresentados para obtenção de limites inferiores para o PTVEP foram avaliados. São eles:

RL Relaxação linear do modelo original.

M1 Modelo para obtenção do ILB, descrito na seção 4.2.1.

M2 Modelo simplificado para obtenção do ILB, descrito na seção 4.2.2.

M3 Modelo para obtenção do ILB2 (ILB melhorado), descrito na seção 4.2.3.

M4 Modelo simplificado para obtenção do ILB2, descrito na seção 4.2.4.

O experimento realizado avalia os modelos em relação à qualidade do limite obtido e o tempo de execução. Os modelos RL, M1 e M2 foram resolvidos para cada instância de teste. Os modelos M3 e M4 foram resolvidos apenas para as instâncias não balanceadas,

pois os mesmos não apresentam melhoria alguma com relação ao ILB para as instâncias balanceadas.

A Tabela 4.1 apresenta o resultado comparativo entre os limites obtidos. A primeira coluna exibe o nome de cada instância. A segunda coluna mostra o valor do limite de relaxação linear. A terceira coluna mostra o valor do ILB . A quarta coluna mostra o valor do ILB2. A quinta coluna mostra a melhoria percentual do valor do ILB em relação ao limite de relaxação linear. A última coluna mostra a melhoria percentual do valor do ILB2 em relação ao ILB.

O ILB superou o limite de relaxação linear para todas as instâncias de testes consideradas, com uma melhoria média de 1,67%. Para as instâncias não balanceadas, o ILB2 apresentou uma melhoria média de 1,30% sobre o valor do ILB.

A melhora no valor do limite do ILB2 com relação ao ILB se deve à adição do lado esquerdo da restrição (4.27). Uma instância desbalanceada pode possuir uma ou mais equipes realizando muitos jogos em casa. Nesse caso, o valor da constante $\lceil \frac{c_i}{3} \rceil - 1$, para a equipe i que realiza muitas partidas em casa, é alto. Isso faz com que a equipe i possua muitas variáveis associadas diferentes de zero, provocando um aumento no número de viagens.

A Tabela 4.2 mostra o resultado comparativo entre os modelos $M1$ e $M2$ para obtenção do ILB com relação ao tempo de execução. A primeira coluna mostra o nome de cada instância. A segunda e terceira colunas mostram os tempos de execução, em unidade de segundos, gastos para solucionar os modelos $M1$ e $M2$, respectivamente. A última coluna mostra o quociente entre os tempos de execução dos modelos $M1$ e $M2$.

O modelo $M2$ foi resolvido mais rápido do que o modelo $M1$ para todas as instâncias. O tempo médio de execução para o modelo $M1$ foi aproximadamente 9 vezes maior do que o tempo médio de execução para o modelo $M2$.

A Tabela 4.3 mostra o resultado comparativo entre os modelos $M3$ e $M4$, para obtenção do ILB melhorado, com relação ao tempo de execução. A primeira coluna mostra o nome de cada instância. A segunda e terceira colunas mostram os tempos de execução, em unidade de segundos, gastos para solucionar os modelos $M3$ e $M4$ respectivamente. A última coluna mostra o quociente entre os tempos de execução dos modelos $M3$ e $M4$.

O modelo $M4$ foi resolvido mais rápido do que o modelo $M3$ para todas as instâncias. O tempo médio de execução para o modelo $M3$ foi aproximadamente 27 vezes maior do que o tempo médio de execução para o modelo $M4$.

A Tabela 4.4 apresenta os tempos de execução para obtenção de cada limite, considerando o melhor modelo para obtenção do ILB (modelo M2) e o melhor modelo para obtenção do ILB2 (modelo M4). A primeira coluna mostra o nome de cada instância. A segunda coluna mostra o tempo de execução para obtenção do limite de relaxação linear. A terceira coluna mostra tempo de execução para obtenção do ILB. A quarta coluna mostra o tempo

Instâncias	RL	ILB	ILB2	ILB x RL (%)	ILB2 x ILB (%)
circ18abal	652	668		2,40	
circ18bbal	645	662		2,58	
circ18cbal	644	660		2,43	
circ18dbal	645	650		0,72	
circ18ebal	643	650		1,09	
circ18fbal	657	668		1,62	
circ18gbal	646	652		0,88	
circ18hbal	649	664		2,36	
circ18ibal	650	658		1,18	
circ18jbal	640	654		2,19	
circ18anonbal	643	666	668	3,63	0,30
circ18dnonbal	654	664	668	1,48	0,60
circ18enonbal	653	666	670	2,04	0,60
circ18fnonbal	643	652	666	1,45	2,15
circ18gnonbal	655	668	672	1,93	0,60
circ18hnonbal	641	650	666	1,46	2,46
circ18inonbal	659	672	684	1,97	1,79
circ18jnonbal	648	656	660	1,29	0,61
circ20abal	869	882		1,50	
circ20bbal	866	874		0,96	
circ20cbal	865	876		1,23	
circ20dbal	867	880		1,46	
circ20ebal	867	888		2,46	
circ20fbal	863	874		1,29	
circ20gbal	863	876		1,55	
circ20hbal	879	886		0,80	
circ20ibal	866	880		1,66	
circ20jbal	866	884		2,04	
circ20anonbal	879	894	906	1,71	1,34
circ20bnonbal	867	880	896	1,46	1,82
circ20cnonbal	867	884	896	1,96	1,36
circ20dnonbal	872	892	914	2,25	2,47
circ20enonbal	873	888	902	1,72	1,58
circ20gnonbal	863	874	896	1,31	2,52
circ20inonbal	871	874	874	0,38	0,00
circ20jnonbal	862	876	882	1,58	0,68
Média	758,13	770,61	782,50	1,67	1,30

Tabela 4.1: Resultado comparativo entre os limites de relaxação linear, ILB e ILB2.

Instâncias	ILB M1	ILB M2	M1/M2
circ18abal	1,632	0,208	7,846
circ18bbal	1,644	0,220	7,473
circ18cbal	1,684	0,192	8,771
circ18dbal	1,632	0,180	9,067
circ18ebal	1,652	0,212	7,792
circ18fbal	1,696	0,204	8,314
circ18gbal	1,688	0,200	8,440
circ18hbal	1,668	0,224	7,446
circ18ibal	1,712	0,180	9,511
circ18jbal	1,648	0,188	8,766
circ18anonbal	1,840	0,252	7,302
circ18dnonbal	2,032	0,284	7,155
circ18enonbal	2,044	0,248	8,242
circ18fnonbal	1,988	0,196	10,143
circ18gnonbal	1,968	0,200	9,840
circ18hnonbal	1,956	0,232	8,431
circ18inonbal	2,144	0,244	8,787
circ18jnonbal	1,784	0,252	7,079
circ20abal	3,044	0,272	11,191
circ20bbal	2,976	0,296	10,054
circ20cbal	3,136	0,292	10,740
circ20dbal	3,164	0,312	10,141
circ20ebal	3,312	0,324	10,222
circ20fbal	3,076	0,280	10,986
circ20gbal	3,080	0,296	10,405
circ20hbal	3,220	0,272	11,838
circ20ibal	3,036	0,316	9,608
circ20jbal	3,040	0,336	9,048
circ20anonbal	3,904	0,296	13,189
circ20bnonbal	3,460	0,320	10,813
circ20cnonbal	3,796	0,352	10,784
circ20dnonbal	3,472	0,400	8,680
circ20enonbal	3,540	0,328	10,793
circ20gnonbal	3,712	0,328	11,317
circ20inonbal	3,484	0,300	11,613
circ20jnonbal	3,624	0,388	9,340
Média	2,569	0,267	9,477

Tabela 4.2: Resultado comparativo entre os tempos dos modelos M1 e M2 para obtenção do ILB.

Instâncias	ILB2 M3	ILB2 M4	M3/M4
circ18anonbal	6,06	0,25	24,03
circ18dnonbal	6,22	0,26	24,28
circ18enonbal	6,20	0,27	23,12
circ18fnonbal	6,07	0,26	23,35
circ18gnonbal	6,26	0,25	24,86
circ18hnonbal	5,88	0,24	24,08
circ18inonbal	6,63	0,32	20,99
circ18jnonbal	5,42	0,24	22,95
circ20anonbal	12,12	0,37	32,93
circ20bnonbal	10,51	0,36	29,52
circ20cnonbal	13,73	0,40	33,98
circ20dnonbal	11,36	0,41	27,83
circ20enonbal	10,66	0,42	25,63
circ20gnonbal	11,03	0,37	29,97
circ20inonbal	10,10	0,35	28,69
circ20jnonbal	10,57	0,34	30,73
Média	8,68	0,32	26,68

Tabela 4.3: Resultado comparativo entre os tempos dos modelos M3 e M4 para obtenção do ILB2.

de execução para obtenção do ILB2. A quinta coluna mostra o quociente entre os tempos de execução para obtenção do limite de relaxação linear e o ILB. A última coluna mostra o quociente entre os tempos de execução para para obtenção do ILB2 e ILB.

O tempo de execução para obtenção do ILB foi muito mais curto do que o tempo de execução para obtenção do limite de relaxação linear para todas as instâncias. O tempo médio para obtenção do limite de relaxação linear foi aproximadamente 959 vezes maior do que o tempo médio para obtenção do ILB. Os resultados indicam que o ILB supera a relaxação linear, tanto com relação à qualidade do limite quanto ao tempo de execução dos respectivos modelos.

Para as instâncias não balanceadas, o tempo de execução para obtenção do ILB2 foi, na média, 11% maior do que o tempo de execução para obtenção do ILB. Em contrapartida, o valor do limite fornecido foi 1,30% melhor.

Instâncias	RL	ILB	ILB2	RL/ILB	ILB2/ILB
circ18abal	129	0,21		620	
circ18bbal	134	0,22		610	
circ18cbal	120	0,19		622	
circ18dbal	141	0,18		783	
circ18ebal	139	0,21		654	
circ18fbal	112	0,20		550	
circ18gbal	122	0,20		608	
circ18hbal	168	0,22		748	
circ18ibal	130	0,18		720	
circ18jbal	149	0,19		790	
circ18anonbal	145	0,25	0,25	574	1,00
circ18dnonbal	122	0,28	0,26	428	0,90
circ18enonbal	178	0,25	0,27	718	1,08
circ18fnonbal	164	0,20	0,26	838	1,33
circ18gnonbal	146	0,20	0,25	728	1,26
circ18hnonbal	138	0,23	0,24	594	1,05
circ18inonbal	174	0,24	0,32	714	1,30
circ18jnonbal	160	0,25	0,24	633	0,94
circ20abal	344	0,27		1264	
circ20bbal	361	0,30		1218	
circ20cbal	374	0,29		1281	
circ20dbal	441	0,31		1412	
circ20ebal	357	0,32		1103	
circ20fbal	285	0,28		1018	
circ20gbal	308	0,30		1040	
circ20hbal	471	0,27		1733	
circ20ibal	413	0,32		1308	
circ20jbal	417	0,34		1241	
circ20anonbal	435	0,30	0,37	1468	1,24
circ20bnonbal	437	0,32	0,36	1366	1,11
circ20cnonbal	431	0,35	0,40	1224	1,15
circ20dnonbal	527	0,40	0,41	1317	1,02
circ20enonbal	359	0,33	0,42	1094	1,27
circ20gnonbal	390	0,33	0,37	1190	1,12
circ20inonbal	435	0,30	0,35	1449	1,17
circ20jnonbal	334	0,39	0,34	861	0,89
Média	269	0,27	0,32	959	1,11

Tabela 4.4: Resultado comparativo com relação ao tempo de execução para obtenção dos limites duais de relaxação linear, ILB e ILB2.

Capítulo 5

Conclusões e Trabalhos Futuros

Este trabalho abordou o Problema do Torneio com Viagens com Estádios Predefinidos. Na primeira parte do trabalho, foi proposta uma heurística de busca local iterada capaz de encontrar boas soluções viáveis para instâncias com tamanhos de interesse prático (até 20 equipes). Uma estratégia generalizada para construção de soluções iniciais foi proposta e avaliada. Mostrou-se que a 1-fatoração canônica não deve ser utilizada para construção de soluções iniciais em algumas situações, onde as 1-fatorações obtidas pelo método generalizado são indicadas. Duas vizinhanças foram investigadas pela heurística de busca local iterada. Mostrou-se que para algumas instâncias, o espaço de soluções é desconectado pelas vizinhanças consideradas.

A nova heurística proposta claramente supera as heurísticas anteriormente conhecidas na literatura, melhorando a melhor solução conhecida anteriormente, para cada instância de teste, em pelo menos 11,42%, com apenas um segundo de execução. A redução média sobre todas as instâncias viáveis foi de 17,17%. Mostrou-se que resultados melhores podem ser obtidos se um limite maior de tempo de execução for aceito.

Na segunda parte do trabalho, foram propostos novos métodos para obtenção de limites inferiores para o Problema do Torneio com Viagens com Estádios Predefinidos. Os limites foram obtidos por meio da resolução de relaxações combinatórias do modelo de programação linear inteira de [Melo, 2007] e [Easton et al., 2001]. Os resultados obtidos superaram os limites de relaxação linear de [Melo, 2007] para todas as instâncias de testes, com um tempo de execução muito menor do que aquele gasto para obter a relaxação linear do modelo.

A Tabela 5.1 resume os resultados numéricos obtidos neste trabalho e compara com os melhores resultados anteriormente conhecidos na literatura. A primeira coluna mostra o nome da instância; a segunda, o valor da melhor solução viável encontrada; a terceira, o melhor limite inferior encontrado. A quarta coluna mostra o GAP de dualidade, i.e. diferença percentual entre o valor da melhor solução viável encontrada e o limite inferior. A quinta coluna mostra o GAP de dualidade de [Melo, 2007]. A última coluna mostra a melhoria

(i.e. redução) percentual comparando os GAPs de dualidade deste trabalho com os GAPs de dualidade de [Melo, 2007].

O GAP de dualidade médio encontrado neste trabalho para as instâncias consideradas foi de aproximadamente 21%, o que representa uma melhoria média de aproximadamente 72% em relação aos resultados anteriormente conhecidos na literatura. Embora a melhoria tenha sido significativa, os GAPs de dualidade encontrados neste trabalho ainda são altos. Há ainda muito o que se investigar em relação ao problema, tanto na obtenção de limites primais quanto na obtenção de limites duais.

Com relação à obtenção de soluções viáveis, uma proposta de trabalho futuro seria investigar a conectividade do espaço de busca e tentar descrever um conjunto de soluções iniciais que permita a heurística alcançar qualquer solução no espaço de soluções. Outro aspecto que pode ser investigado é a criação de novas vizinhanças que possam conectar o espaço de busca.

Em relação à obtenção de limites duais, uma proposta de trabalho futuro seria investigar o uso de uma relaxação lagrangiana do modelo de programação inteira de [Melo, 2007]. A relaxação combinatória do modelo suprimindo o conjunto de restrições (4.4) permitiu duas simplificações: (i) decomposição do modelo por equipes e (ii) redução do número de variáveis. Isso fez com que o modelo relaxado pudesse ser resolvido rapidamente. Como a solução do modelo relaxado foi melhor do que o limite de relaxação linear do modelo original, acredita-se que exista uma possibilidade de melhorar o limite obtido, dualizando as restrições relaxadas (4.4) e as penalizando na função objetivo. A solução da relaxação para o cálculo do ILB melhorado equivale a uma solução da relaxação lagrangiana com todos os multiplicadores nulos. Acredita-se que possa existir um outro conjunto de multiplicadores que forneça um limite melhor do que o ILB melhorado.

Instância	Primal	Dual	GAP (%)	GAP (%) de Melo [2007]	Melhoria
circ18abal	776	668	16,17	69,37	76,01
circ18bbal	796	662	20,24	70,28	71,48
circ18cbal	794	660	20,30	60,93	58,04
circ18dbal	788	650	21,23	69,66	69,18
circ18ebal	784	650	20,62	67,03	66,31
circ18fbal	792	668	18,56	61,09	60,76
circ18gbal	784	652	20,25	70,02	71,10
circ18hbal	780	664	17,47	68,57	73,00
circ18ibal	778	658	18,24	69,28	72,92
circ18jbal	780	654	19,27	68,44	70,24
circ18anonbal	806	668	20,66	74,81	77,35
circ18dnonbal	800	668	19,76	61,83	60,10
circ18enonbal	804	670	20,00	67,23	67,47
circ18fnonbal	802	666	20,42	70,76	71,92
circ18gnonbal	782	672	16,37	67,38	72,87
circ18hnonbal	804	666	20,72	73,17	74,92
circ18inonbal	818	684	19,59	67,53	68,48
circ18jnonbal	782	660	18,48	70,06	73,68
circ20abal	1094	882	24,04	74,91	72,68
circ20bbal	1082	874	23,80	76,67	75,54
circ20cbal	1072	876	22,37	69,75	67,67
circ20dbal	1100	880	25,00	68,66	62,38
circ20ebal	1076	888	21,17	76,01	78,34
circ20fbal	1072	874	22,65	79,14	80,70
circ20gbal	1068	876	21,92	77,98	80,09
circ20hbal	1094	886	23,48	72,47	69,99
circ20ibal	1078	880	22,50	78,29	79,70
circ20jbal	1086	884	22,85	71,16	69,02
circ20anonbal	1106	906	22,08	70,88	69,72
circ20bnonbal	1082	896	20,76	75,35	77,98
circ20cnonbal	1096	896	22,32	71,63	70,44
circ20dnonbal	1136	914	24,29	72,97	69,54
circ20enonbal	1100	902	21,95	80,30	83,35
circ20gnonbal	1078	896	20,31	78,45	83,05
circ20inonbal	1082	874	23,80	74,05	71,79
circ20jnonbal	1070	882	21,32	75,67	77,64
Média:	939,50	775,17	20,97	71,44	72,10

Tabela 5.1: Comparação entre os GAPs de dualidade deste trabalho com os resultados anteriores da literatura.

Referências Bibliográficas

Anagnostopoulos, A.; Michel, L.; Hentenryck, P. V. e Vergados, Y. (2006). A simulated annealing approach to the traveling tournament problem. *Journal of Scheduling*, 9:177–193.

Benoist, T.; Laburthe, F. e Rottembourg, B. (2001). Lagrange relaxation and constraint programming collaborative schemes for travelling tournament problems. In *Proceedings of the Third International Workshop on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR'01)*, Ashford.

Bondy, J. A. e Murty, U. S. R. (1979). *Graph Theory with Applications*. Elsevier, New York.

Briskorn, D. (2006). Scheduling sport leagues using branch-and-price. In Burke, E. e Rudová, H., editores, *Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling*, pp. 367–369, Brno.

Croce, F. D. e Oliveri, D. (2006). Scheduling the Italian football league: an ILP-based approach. *Computers and Operations Research*, 33:1963–1974.

Dantzig, G. B. e Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6(1):80–91.

de Werra, D. (1980). Geography, games, and graphs. *Discrete Applied Mathematics*, 2:327–337.

de Werra, D. (1981). Scheduling in sports. In Hansen, P., editor, *Studies on Graphs and Discrete Programming*, volume 11 of *Annals of Discrete Mathematics*, pp. 381–395. North Holland.

Dinitz, J. H.; Garnick, D. K. e Mckay, B. D. (1994). There are 526,915,620 nonisomorphic one-factorizations of k12. *Journal of Combinatorial Design*, 2:273–285.

Easton, K.; Nemhauser, G. e Trick, M. (2001). The traveling tournament problem: Description and benchmarks. In Walsh, T., editor, *Principles and Practice of Constraint Programming*, volume 2239 of *Lecture Notes in Computer Science*, pp. 580–584. Springer.

Easton, K.; Nemhauser, G. e Trick, M. (2003). Solving the traveling tournament problem: A combined integer programming and constraint programming approach. In Goos, G.; Hartmanis, J. e van Leeuwen, J., editores, *Practice and Theory of Automated Timetabling IV*, volume 2740 of *Lecture Notes in Computer Science*, pp. 100–109. Springer.

Easton, K.; Nemhauser, G. e Trick, M. (2004). Sports scheduling. In Leung, J., editor, *Handbook of Scheduling: Algorithms, Models and Performance Analysis*, pp. 52.1–52.19. CRC Press.

Gaspero, D. e Schaerf, A. (2007). A composite-neighborhood tabu search approach to the traveling tournament problem. *Journal of Heuristics*, 13:189–207.

Glover, F. e Kochenberger, G. (2003). *Handbook of Metaheuristics*. Springer.

Goossens, D. e Spieksma, F. (2006). Scheduling the Belgian soccer league. In Burke, E. e Rudová, H., editores, *Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling*, pp. 420–422, Brno.

Hentenryck, V. e Vergados, Y. (2005). Minimizing breaks in sport scheduling with local search. In Biundo, S.; Myers, K. e Rajan, K., editores, *Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS 2005)*, pp. 22–29, Monterey.

Henz, M. (1999). Constraint-based round robin tournament planning. In Schreye, D. D., editor, *International Conference on Logic Programming*, pp. 545–557, New Mexico.

Henz, M.; Müller, T. e Thiel, S. (2004). Global constraints for round robin tournament scheduling. *European Journal of Operational Research*, 153:92–101.

Hoos, H. e Stuzle, T. (2005). *Stochastic Local Search Foundations and Applications*. Morgan, Kaufmann.

Kendall, G.; Knust, S.; Ribeiro, C. e Urrutia, S. (2008). Scheduling in sports: An annotated bibliography. Submetido para publicação.

Lourenço, H. R.; Martin, O. C. e Stutzle, T. (2003). Iterated local search. In Glover, F. e Kochenberger, G., editores, *Handbook of Metaheuristics*. Springer.

Melo, R. A. (2007). Modelos de programação inteira para o problema do torneio com viagens com estádios fixos. Dissertação de mestrado, Universidade Federal Fluminense, Brasil.

Nemhauser, G. e Trick, M. (1998). Scheduling a major college basketball conference. *Operations Research*, 46:1–8.

Rasmussen, R. (2008). Scheduling a triple round robin tournament for the best Danish soccer league. *European Journal of Operational Research*, 185:795–810.

Rasmussen, R. e Trick, M. (2006). The timetable constrained distance minimization problem. In Beck, J. e Smith, B., editores, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 3990 of *Lecture Notes in Computer Science*, pp. 167–181. Springer.

Rasmussen, R. e Trick, M. (2008). Round robin scheduling - A survey. *European Journal of Operational Research*, 188:617–636.

Régin, J. (2001). Minimization of the number of breaks in sports scheduling problems using constraint programming. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 57:115–130.

Ribeiro, C. C. e Urrutia, S. (2007a). Heuristics for the mirrored traveling tournament problem. *European Journal of Operational Research*, 179:775–787.

Ribeiro, C. C. e Urrutia, S. (2007b). Scheduling the Brazilian soccer championship. In Burke, E. e Rudová, H., editores, *Practice and Theory of Automated Timetabling VI*, volume 3867 of *Lecture Notes in Computer Science*, pp. 149–159. Springer.

Russell, R. e Urban, T. (2006). A constraint programming approach to the multiple-venue, sport-scheduling problem. *Computers and Operations Research*, 33:1895–1906.

Trick, M. (2009). Challenge traveling tournament instances. `http://mat.gsia.cmu.edu/TOURN/`, última visita em março de 2009.

Urrutia, S.; Ribeiro, C. e Melo, R. (2007). A new lower bound to the traveling tournament problem. In *Computational Intelligence in Scheduling*, pp. 15–18.