PROGRAMAÇÃO DE TABELA DE JOGOS PARA CAMPEONATOS POR MEIO DA METAHEURÍSTICA *ITERADED LOCAL SEARCH*

Igor Andrade Lessa de Oliveira[1] **, Tárik Lemos Reis Porto**[1]

[1]Departamento de Computação – Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG)

Av. Amazonas 5253 - Nova Suíça - Belo Horizonte - MG - Brasil CEP: 30.421-169

Igor.lessa08@gmail.com, tariklemos1511@gmail.com

Abstract. This paper intends to describe a heuristic solution based on *Iterated Local Search* for the *Travelling Tournament Problem* (TTP). The main objective of this solution is to reduce the overall amount of distance traveled during the competition. In this work is explained the importance of these kind of practices while managing high performance tournaments.

Resumo. Esse artigo pretende descrever uma solução heurística baseada no algoritmo de *Iterated Local Search* para o *Travelling Tournment Problem* (TTP). O objetivo principal dessa solução é reduzir a distância total percorrida durante competições. Nesse artigo é explicado a importância desse tipo de prática no gerenciamento de competições de alta performance.

1. Introdução

As competições esportivas despertam atenção de vários grupos de pessoas. São eventos que, além de cumprirem seu papel social como forma de entretenimento, desempenho atlético e inclusão, também se tornaram um ótimo palco para investimentos, atraindo diversos patrocinadores e investidores, para diversas modalidades esportivas. No que diz respeito a práticas de alto desempenho, como em competições oficiais com atletas de alto nível, se tem conhecimento de que qualquer detalhe pode influenciar na performance de um competidor. Tendo isso em vista, é importante minimizar gastos comuns entre diferentes atletas, como por exemplo, as horas de viagem entre uma atividade e outra, até mesmo como forma de promover a competitividade. Entretanto, como atualmente as competições movimentam muito dinheiro e são vistas como verdadeiros investimentos econômicos, é vital que se busque uma redução de custos, para maximizar os lucros. Um dos gastos em competições que podem ser reduzidos é o de transporte, para isso é necessário que se monte tabelas mais adequadas, e métodos que analisem a distância percorrida ao longo da competição e tentem diminuir esse valor.

A alocação de ótima de jogos é uma tarefa muito complexa, pois se enquadra na classe de problemas NP-difíceis, sendo, portanto um problema altamente combinatório. Tomando como exemplo o Campeonato Brasileiro de Futebol, no qual estão presentes 20 times, o número de combinações possíveis de tabelas é de 2,9062 x 10^{130} e pode ser obtido pela fórmula:

$$(n-1)!\,(n-3)!\,(n-5)!\,\ldots\big(n-(n-1)\big)!\,2^{(n-1)\frac{n}{2}}$$

Com um número tão grande de possibilidades, a resolução desse problema por métodos exatos se torna impraticável. Sendo assim, é necessário que se busque técnicas aproximadas ou heurísticas, capazes de obter uma solução satisfatória ao sistema. Nas seções seguintes desse artigo descreveremos uma meta-heurística para resolver esse problema, utilizando

ITERADED LOCAL SEARCH. O método utilizado pode ser migrado para qualquer outra competição, desde que seja adequado às regras da mesma.

2. Descrição do problema

O problema da programação de jogos de competições esportivas realizadas em dois turnos espelhados (PPJE) é encontrada em diversas competições no âmbito nacional, como por exemplo no Campeonato Brasil de Futebol, no Novo Basquete Brasil, Super Liga de Vôlei, entre outros. A situação é basicamente a seguinte: A competição é dividida em dois turnos, cada um com n-1 rodadas, sendo n o número de times inscritos no torneio. Todos os times se enfrentam pelo menos uma vez por turno, sendo que, caso a partida em um turno seja em casa, no outro turno ela tem que ser obrigatoriamente, fora de casa. Quando um time jogar duas partidas consecutivas fora de casa, assume-se que ele sairá da cidade do primeiro oponente diretamente para a cidade do segundo, sem retornar à sua sede. A tabela 1 abaixo, representa um modelo válido a essas condições com 6 times inscritos.

Tabela 1: Exemplo de uma escala de jogos do PPJE

Rodadas									
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1 x 5	3 x 2	4 x 6	5 x 6	5 x 3	5 x 1	2 x 3	6 x 4	6 x 5	3 x 5
4 x 2	1 x 6	2 x 5	2 x 1	6 x 2	2 x 4	6 x 1	5 x 2	1 x 2	2 x 6
6 x 3	5 x 4	3 x 1	4 x 3	1 x 4	3 x 6	4 x 5	1 x 3	3 x 4	4 x 1
1º Turno					**2º Turno**				

A tabela 1 ilustra claramente a situação, as 5 primeiras colunas representam o primeiro turno, enquanto as outras o segundo turno. Times que estão jogando em casa estão representados pelo fato de aparecerem à esquerda na cédula da partida. Pode se perceber também como os mandos de campo são invertidos no segundo turno.

Na competição que simulamos nos testes realizados, temos 8 times inscritos, cada um de um estado diferente, totalizando 14 rodadas. As restrições da competição simulada foram as seguintes:

a) Cada time joga somente uma vez por rodada;

b) Dois times jogarão entre si duas vezes, uma no turno e a outra no returno, alternando o mando de campo entre os mesmos;

c) A diferença entre os jogos feitos em cada turno em casa e fora de casa de um time não pode ser maior que uma unidade;

d) Um time não pode jogar mais que duas vezes consecutivas dentro ou fora de casa.

O objetivo principal do PPJE é minimizar os custos com transporte na competição, ou seja, diminuir a distância total percorrida durante o torneio. O objetivo secundário é minimizar a diferença entre o time que mais percorreu e o que menos percorreu. Em seguida descreveremos a metodologia utilizada no processo.

3. Metodologia

3.1 Representação de uma solução

Seja *n* o número de times inscritos na competição, e, $nr = 2n - 2$ o número total de rodadas do torneio, temos que, qualquer solução para o problema tem de estar no formato de uma matriz de *n* linhas por *nr* colunas. Cada linha representa a participação de um time na competição, as cédulas dessa linha indica o adversário da rodada referente à coluna da cédula. No caso do sinal na cédula ser negativo, o time representada na linha jogará fora de casa, em caso de positivo, jogará em casa. A tabela 2 exemplifica uma solução, considerando 6 times e 10 rodadas.

Tabela 2: Exemplo de solução na representação de Anagnostopoulos *et al.* (2003)

Time \ Rodada	1	2	3	4	5	6	7	8	9	10
1	+ 6	- 5	+ 4	+ 3	- 2	- 4	- 3	+ 2	+ 5	- 6
2	+ 5	- 3	+ 6	+ 4	+ 1	- 6	- 4	- 1	+ 3	- 5
3	- 4	+ 2	+ 5	- 1	+ 6	- 5	+ 1	- 6	- 2	+ 4
4	+ 3	+ 6	- 1	- 2	- 5	+ 1	+ 2	+ 5	- 6	- 3
5	- 2	+ 1	- 3	- 6	+ 4	+ 3	+ 6	- 4	- 1	+ 2
6	- 1	- 4	- 2	+ 5	- 3	+ 2	- 5	+ 3	+ 4	+ 1

Na Tabela 2, observa-se, por exemplo, que a notação +5, encontrada na primeira rodada do time 2, indica que o time 2 joga em sua sede contra o time 5, enquanto que na rodada 4 do time 3, encontra-se a notação -1, cujo significado é que o time 3 joga fora de sua sede (no caso, na casa do oponente) contra o time 1.

Essa representação nos permite definir claramente a rota que um time irá percorrer durante a competição, pois essa estará toda descrita em sua linha referente.

Trajetória do time 6: $6 \xrightarrow{1} 1 \xrightarrow{2} 4 \xrightarrow{3} 2 \xrightarrow{4} 6 \xrightarrow{5} 3 \xrightarrow{6} 6 \xrightarrow{7} 5 \xrightarrow{8} 6$

Trajetória do time 2: $2 \xrightarrow{2} 3 \xrightarrow{3} 2 \xrightarrow{6} 6 \xrightarrow{7} 4 \xrightarrow{8} 1 \xrightarrow{9} 2 \xrightarrow{10} 5 \mapsto 2$

3.2 Vizinhança de uma solução

As soluções vizinhas são geradas por meio das funções swap rounds e swap homes, ambas utilizadas por Anagnostopoulos et al. (2003). O comando swap rounds consiste em trocar os jogos de duas colunas entre si, por exemplo, usando esse movimento para trocar os jogos das rodadas 3 e 7 da tabela 2, o resultado seria o apresentado na Tabela 3.

Tabela 3: Solução vizinha gerada pelo movimento *swap rounds*

Time \ Rodada	1	2	3	4	5	6	7	8	9	10
1	+ 6	- 5	- 3	+ 3	- 2	- 4	+ 4	+ 2	+ 5	- 6
2	+ 5	- 3	- 4	+ 4	+ 1	- 6	+ 6	- 1	+ 3	- 5
3	- 4	+ 2	+ 1	- 1	+ 6	- 5	+ 5	- 6	- 2	+ 4
4	+ 3	+ 6	+ 2	- 2	- 5	+ 1	- 1	+ 5	- 6	- 3
5	- 2	+ 1	+ 6	- 6	+ 4	+ 3	- 3	- 4	- 1	+ 2
6	- 1	- 4	- 5	+ 5	- 3	+ 2	- 2	+ 3	+ 4	+ 1

O comando swap homes consiste em trocar o mando de campo entre os jogos de dois times, por exemplo, usando esse movimento para os jogos dos times 1 e 4 da Tabela 2, o resultado seria o da Tabela 4.

Time \ Rodada	1	2	3	4	5	6	7	8	9	10
1	+ 6	- 5	+ 4	+ 3	- 2	- 6	+ 5	- 4	- 3	+ 2
2	+ 5	- 3	+ 6	+ 4	+ 1	- 5	+ 3	- 6	- 4	- 1
3	- 4	+ 2	+ 5	- 1	+ 6	+ 4	- 2	- 5	+ 1	- 6
4	+ 3	+ 6	- 1	- 2	- 5	- 3	- 6	+ 1	+ 2	+ 5
5	- 2	+ 1	- 3	- 6	+ 4	+ 2	- 1	+ 3	+ 6	- 4
6	- 1	- 4	- 2	+ 5	- 3	+ 1	+ 4	+ 2	- 5	+ 3

Caso as soluções resultantes dos movimentos não respeitem as condições necessárias, o movimento é descartado.

3.3 Função de Avaliação

Uma solução S, é avaliada por uma função f de acordo com a fórmula.

$$f(S) = \sum_{i=1}^{n} \sum_{n=1}^{nr+1} distancia(n, nr)$$

Ou seja, é feito um somatório da distância percorrida por cada time em cada rodada. A função $distancia(n, nr)$ retorna a distância percorrida pelo time da linha *n*, na rodada corrente, o somatório tem de ir até *nr+1*, pois ao final da última rodada os times retornam à sua sede principal, o que é interpretado como um deslocamento de uma rodada extra. As soluções que apresentarem um valor menor para esse somatório serão as escolhidas para atingir o objetivo principal. Pois isso indica que nelas a distância percorrida pelos times foi menor, e que houve redução de custo.

3.4 Geração de uma Solução Inicial

A geração da solução inicial foi realizada sem a utilização de um método específico, em nossa implementação utilizamos como inicial a tabela 5 a baixo. Existem formas de se buscar uma solução inicial mais satisfatório, como por exemplo utilizando *backtracking.*

Tabela 5: Solução Inicial utilizada.

Time \ Rodada	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	+4	-2	+3	-5	-7	-6	+8	-4	+2	-3	+5	+7	+6	-8
2	-5	+1	+4	+6	-8	+7	-3	+5	-1	-4	-6	+8	-7	+3
3	+6	-4	-1	+7	+5	-8	+2	-6	+4	+1	-7	-5	+8	-2
4	-1	+3	-2	+8	-6	+5	-7	+1	-3	+2	-8	+6	-5	+7
5	+2	-8	+7	+1	-3	-4	+6	-2	+8	-7	-1	+3	+4	-6
6	-3	+7	-8	-2	+4	+1	-5	+3	-7	+8	+2	-4	-1	+5
7	+8	-6	-5	-3	+1	-2	+4	-8	+6	+5	+3	-1	+2	-4
8	-7	+5	+6	-4	+2	+3	-1	+7	-5	-6	+4	-2	-3	+1

3.5 *Iterated Local Search* Aplicada ao PPJE

O refinamento da solução inicial descrita no item anterior, foi feito tanto utilizando um método hibrido, ILS – MRD, composto pela metaheurística Busca Local Iterado (ILS) e pelo Método Randômico de Descida (MRD), sendo este usado como um mecanismo de busca local para o ILS. O pseudocódigo do procedimento adaptado ao PPJE está representado na figura 1 abaixo.

```
procedimento ILS-MRD
    s0 ← SolucaoInicial.
    s ← MRD(s0, IterMRD)
    kp ← kp0
    iter ← 0
    enquanto (kp < kpmax)
        enquanto (iter ← melhorIter < itermax)
            iter ← iter + 1
            s' ← pertubação(s, kp)
            s" ← MRD(s', IterMRD)
            se ( f(s") < f(s) ) faça
                s ← s"
                melhorIter ← iter
                kp ← kp0
            fim-se
        fim-enquanto
        kp ← kp + delta
    fim-enquanto
    retorne s
```

Figura 1: Algoritmo ILS-MRD

O algoritmo parte de uma solução inicial já definida, e aplica um mecanismo de busca local, o Método Randômico de Descida (MRD). O método funciona com a escolha aleatória de um tipo de movimento, entre os apresentados na seção 3.2 e a partir dessa escolha é gerado um vizinho aleatório. É feita uma avaliação para determinar se o vizinho gerado é melhor que o corrente, se for, ele passa a ser a nova solução corrente. Caso contrário, um novo vizinho é gerado. O método MRD possui um ponto de parada quando o número máximo de iterações sem melhora (I*terMRD*) for atingido. A cada iteração do método ILS-MRD, gera-se uma perturbação na solução corrente, que consiste na realização de *k* movimentos em uma estrutura de vizinhança escolhida aleatoriamente. Sendo k um número entre 1 e *kp.* A essa nova solução vizinha s’, se aplica o MRD, gerando uma solução vizinha ótima local s’’. Se s’’ for melhor que s corrente, então s’’ se torna a solução corrente, e o valor de *kp* é resetado. O processo se repete até que o número de iterações sem sucesso realizadas seja atingido. Quando isso ocorrer, incrementa-se o grau de perturbação, *kp*, em um fator delta, quando *kp* atingir seu valor máximo (kp_{max}), chegou ao fim.

4. Resultados Computacionais

Apresentam-se aqui os resultados obtidos aplicando-se os métodos ILS-MRD (*Iterated Local Search* e Método Randômico de Descida) e ILS (Iterated Local Search) para resolver o PPJE. O programa foi desenvolvido na linguagem MATLAB utilizando o software MATLAB, versão 7.12.0. O computador utilizado foi um Intel Core i5 2,5 GHz com 6 GB de memória RAM, rodando no sistema operacional Windows 7.

Como objeto de análise usou-se um torneio fictício de 8 equipes e roda-se os algoritmos ILS-MRB e ILS para comparação dos resultados.

Os parâmetros utilizados foram: $kp0 = 1, kpMax = 5, iterMax = 350,$ $delta = 2, \mathrm{I}terMRD = 1000$.

Na tabela 6, mostramos o melhor valor, média, desvio padrão e *tempo médio(TM)* de cem execuções dos algoritmos ILS-MRD e ILS.

Tabela 6: Testes para 100 soluções.

ILS-MRD				**ILS**			
Melhor (km)	Média (km)	Desvio Padrão	TM (s)	Melhor (km)	Média (km)	Desvio	TM (s)
64087	67420,19	5,201039	1013	66282	71821,61	8,357639	452

Ao analisarmos os dados da tabela, podemos perceber que para uma entrada de dados de 8 times, a diferença entre o melhor caso encontrado em todos os testes para cada algoritmo é muito pequena, sendo que o valo encontrado pelo ILS cerca de 1,4% maior que o do ILS-

MRD. Em contra partida, no que diz respeito ao caso médio, a diferença foi somente de 0,06%, o que mostra que ambos os métodos são bem robustos. No que diz respeito ao tempo, o ILS-MRD demorou próximo a duas vezes mais que o método ILS, o que é explicado pelo loop externo no algoritmo. O gráfico 1 abaixo, mostra a dispersão dos testes realizados, a série 1 representa ILS-MRD, enquanto a série 2 representa ILS. Nele é possível perceber a robustez dos métodos, devido a proximidade entre todas as soluções encontradas.

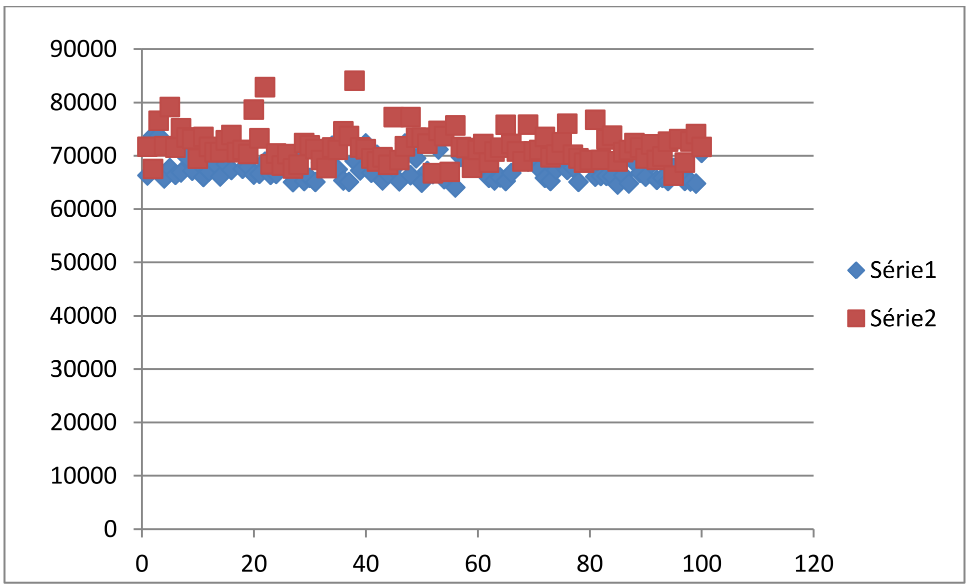

Gráfico 1: Dispersão ILS e ILS-MRD

5. Considerações Finais

O método de ILS-MRD se mostrou extramente eficiente no que diz respeito à obtenção de uma solução heurística para o problema abordado. Nas entradas de dados testados não foi possível perceber uma grande melhora em relação ao método ILS, mas, em problemas com mais times, esperamos que o refinamento seja mas perceptível. O objetivo do método proposto foi atingido, pois a solução encontrada permite reduzir o custo com o transporte, que, ao se tratar de grandes competições, pode chegar a milhões. O método também se mostrou muito robusto, com uma dispersão muito pequena entre todas as soluções encontradas.

6. Referências:

PROGRAMAÇÃO DE JOGOS DA PRIMEIRA DIVISÃO DO CAMPEONATO BRASILEIRO DE FUTEBOL POR MEIO DA METAHEURÍSTICA ITERATED LOCAL SEARCH

<http://www.decom.ufop.br/prof/marcone/Publicacoes/SPOLM-2005-PPJ.pdf> Acesso em 15/02/2014 – Artigo base para a realização desse.

Introdução ao Escalonamento e Aplicações : Escalonamento no Futebol Brasileiro

<http://www.ime.usp.br/~gold/cursos/2009/mac5758/MarcioFutebol.pdf> Acesso em : 15/02/2014