3
Heurísticas para o PAA básico

Como demonstrado no capítulo anterior, a versão de decisão do problema básico de atribuição de árbitros é um problema NP-completo. Também no capítulo anterior, foram apresentadas formulações do PAA por programação inteira que permitiram a resolução exata de instâncias com dimensões pequenas e médias, com até 80 partidas e 160 árbitros. No entanto, instâncias originadas de aplicações reais, que envolvem até 500 partidas e 750 árbitros, não puderam ser resolvidas de maneira exata através daquelas formulações.

Neste capítulo, propõe-se um abordagem heurística de três fases para a resolução aproximada de instâncias com tamanho real. A primeira fase consiste em aplicar uma heurística construtiva aleatorizada para encontrar uma solução inicial, que pode eventualmente violar algumas restrições do problema. A segunda fase é uma heurística reparadora, baseada na metaheurística *Iterated Local Search* (ILS) (50, 52, 53), que é utilizada sempre que necessário para tornar viável a solução inicial. Por fim, uma outra heurística, também baseada em ILS, tem como objetivo aprimorar a solução viável encontrada pela heurística construtiva ou pela reparadora. O Pseudo-código 1 mostra o esquema geral desta abordagem.

```
1  Pseudo-código HeuristicaPAA(MaxIter1, MaxIter2, MaxIterBL)
2  Solucao ← Constroi();
3  if Solucao não é viável then
4      Solucao ← Repara(Solucao, MaxIter1, MaxIterBL);
5  if Solucao é viavel then
6      Solucao ← Aprimora(Solucao, MaxIter2, MaxIterBL);
7      return Solucao;
8  else
9      return Nenhuma solução viável encontrada;
10 end
```

Pseudo-código 1: Heurística de três fases para atribuição de árbitros.

As próximas seções apresentam em detalhes os algoritmos envolvidos em cada fase desta abordagem.

3.1 Heurísticas construtivas

A primeira e a segunda fases da abordagem proposta no Pseudo-código 1 têm como objetivo encontrar uma solução inicial viável. Como já observado por Urrutia (71), pode-se constatar uma melhora significativa no desempenho de heurísticas para problemas combinatórios em esportes quando se compara heurísticas nas quais uma etapa de pré-processamento é empregada para construir boas soluções iniciais com aquelas em que a construção de soluções iniciais é feita de forma aleatória. Além disso, algoritmos construtivos devem ser rápidos, pois freqüentemente são executados repetidas vezes durante o processo de resolução do problema.

São propostas três heurísticas construtivas descritas a seguir.

3.1.1 Heurística construtiva que prioriza a viabilidade

A primeira heurística construtiva para o PAA é denominada `CEV` (Construtivo com Ênfase em Viabilidade) e têm como objetivo a construção de uma solução viável inicial, não levando em consideração o seu custo. A estratégia adotada neste algoritmo tem a preocupação de aumentar as chances de se encontrar uma solução que não viole alguma das restrições do problema, ou que viole o menor número possível.

Uma vez que os árbitros que atuam também como jogadores podem ser designados a arbitrar partidas somente na mesma localidade onde irão jogar, o algoritmo inicia o processo pela tentativa de efetuar a atribuição de tais árbitros ao maior número de PA's que estejam associadas a partidas que acontecem na mesma localidade em que jogam.

Em seguida, a heurística iterativamente: (1) escolhe de forma gulosa a localidade f com maior necessidade por árbitros cujo nível de qualificação seja o máximo ($\overline{p}$) dentre todos os árbitros disponíveis; (2) seleciona aleatoriamente um árbitro disponível com nível de qualificação igual a $\overline{p}$ e (3) mantendo a viabilidade da solução parcialmente preenchida, atribui a este árbitro tantas PA's quanto possível, dentre aqueles associados a partidas que acontecem na localidade f e que ainda não possuem árbitro designado. Neste último passo, fica claro que a prioridade é a viabilidade da solução final, uma vez que o número de partidas que este árbitro deseja arbitrar não é levado em consideração.

Este procedimento é aplicado enquanto houver PA's sem árbitros atribuídos. Ao final, se alguma PA ainda permanecer sem árbitro atribuído, a solução é completada com atribuições inviáveis.

O Pseudo-código 2 ilustra os passos dessa heurística. Denota-se por S^u o conjunto que contém todas as PA's não atribuídas a árbitro algum, por R^{HF} o conjunto de árbitros associados a uma restrição forte quanto à localidade em que devem atuar, e por R^{NHF} o conjunto de árbitros não associados a restrição de localidade alguma, isto é, $R = R^{HF} \cup R^{NHF}$. Estes conjuntos são inicializados respectivamente nas linhas 2, 3 e 4. O laço nas linhas 5 a 14 é executado até que todos os árbitros com restrição de localidade tenham sido examinados e atribuídos a tantas PA's quanto possível. Um árbitro i associado a uma restrição de localidade é aleatoriamente selecionado na linha 6 e removido de R^{HF} na linha 7, $i = 1, \ldots, m$. Sejam $HF(i)$ a localidade onde o árbitro i, que também atua como jogador, irá jogar sua partida e $F(j)$ a localidade onde ocorre a partida associada à PA j. Assim, a localidade f onde o árbitro em questão jogará sua partida é identificada na linha 8. O laço nas linhas 9 a 13 investiga todas as PA's ainda sem árbitro atribuído associadas a partidas que acontecem nesta localidade. Se este árbitro pode ser atribuído a uma PA j, para algum $j = 1, \ldots, n$, sem a violação de alguma restrição (linha 10), então a atribuição é feita (linha 11) e esta PA é removida do conjunto S^u na linha 12.

Em seguida, o laço nas linhas 15 a 30 procura preencher as PA's que permanecem não atribuídas, com árbitros sem restrições de localidade. Na linha 16, $\overline{p}$ é inicializado com o valor do nível de qualificação máximo dentre todos os árbitros em R^{NHF}. A localidade f com a maior necessidade por árbitros com nível de qualificação igual a $\overline{p}$ é selecionada de forma gulosa na linha 17. Um árbitro $i \in R^{NHF}$ com $p_i = \overline{p}$ é selecionado aleatoriamente na linha 18, removido de R^{NHF} na linha 19 e atribuído a tantas PA's quanto possível nas linhas 21 a 29. Na linha 20, atribui-se a F^{tmp} uma cópia do conjunto de localidades, sem a localidade f. O laço nas linhas 22 a 26 investiga todas as PA's sem árbitro atribuído e associadas a partidas que acontecem na localidade f. Se o árbitro i, para algum $i = 1, \ldots, m$, pode ser atribuído a uma PA j, para algum $j = 1, \ldots, n$, sem a violação de alguma restrição (linha 23), então a atribuição é realizada na linha 24 e a PA j é removida de S^u na linha 25. Na linha 27, uma nova localidade é selecionada para o caso em que nenhuma PA da localidade anterior houver sido atribuída ao árbitro em questão. Esta localidade é removida de F^{tmp} na linha 28. Neste caso, o laço das linhas 21 a 29 é novamente executado, agora para a nova localidade selecionada.

O laço nas linhas 31 a 36 completa a solução com atribuições inviáveis, para cada PA $j \in S^u$ ainda sem árbitro atribuído. A localidade f em que ocorre a partida correspondente à PA j é determinada na linha 32. Um árbitro que já esteja designado a pelo menos uma partida nessa localidade (ou não designado

```
1  Pseudo-código CEV()
2  R^HF ← {i ∈ R : i joga pelo menos uma partida};
3  R^NHF ← R\R^HF;
4  S^u ← {j ∈ S : Σ_{i=1}^{m} x_ij = 0};
5  while S^u ≠ ∅ and R^HF ≠ ∅ do
6      Selecionar aleatoriamente i ∈ R^HF;
7      R^HF ← R^HF\{i};
8      f ← HF(i);
9      forall j ∈ S^u : F(j) = f do
10         if a atribuição do árbitro i à PA j é viável then
11             x_ij ← 1;
12             S^u ← S^u\{j};
13     end
14 end
15 while S^u ≠ ∅ and R^NHF ≠ ∅ do
16     p̄ ← max_{i∈R^NHF}{p_i};
17     Seja f a localidade com a maior necessidade por árbitros com
       nível de qualificação igual a p̄;
18     Selecionar aleatoriamente i ∈ R^NHF tal que p_i = p̄;
19     R^NHF ← R^NHF\{i};
20     F^tmp ← F\{f};
21     while o árbitro i não estiver designado a PA alguma do
22         forall j ∈ S^u : F(j) = f do
23             if a atribuição do árbitro i à PA j é viável then
24                 x_ij ← 1;
25                 S^u ← S^u\{j};
26         end
27         Seja f a localidade pertencente a F^tmp com a maior
           necessidade por árbitros com nível de qualificação igual a p̄;
28         F^tmp ← F^tmp\{f};
29     end
30 end
31 forall j ∈ S^u do
32     f ← F(j);
33     Selecionar aleatoriamente i ∈ R : Σ_{j'∈S^u(f)} x_ij' > 0 ou
       [ Σ_{j'=1}^{n} x_ij' = 0 e (i ∉ R^HF ou HF(i) = f ]);
34     x_ij ← 1;
35     S^u ← S^u\{j};
36 end
37 return Solucao : {(i, j) : i ∈ R, j ∈ S e x_ij = 1};
```

Pseudo-código 2: Heurística construtiva aleatorizada com ênfase na viabilidade.

a partida alguma) é selecionado aleatoriamente na linha 33. A atribuição deste árbitro à PA j é realizada na linha 34 e esta PA é removida de S^u na linha

35. A escolha feita na linha 33 garante que todas as partidas atribuídas a um mesmo árbitro ocorrem na mesma localidade.

Um critério guloso é aplicado na linha 17 para selecionar a localidade com a maior necessidade por árbitros com um certo nível de qualificação. Para se entender o critério utilizado, considere o seguinte exemplo: seja uma instância do PAA com uma única localidade denominada a na qual acontecem três partidas g_1, g_2, e g_3, com três posições de arbitragem cada: P_1, P_2, e P_3. A Tabela 3.1 apresenta os valores para os níveis mínimos de qualificação exigidos por cada PA de cada partida. Um "$-$" informa que a posição de arbitragem correspondente não requer um árbitro ou algum árbitro já foi designado a ela. Assim, somente as posições ainda sem árbitro designado são consideradas na determinação do critério guloso.

		Posições de arbitragem		
		P_1	P_2	P_3
Partidas	g_1	4	$-$	3
	g_2	5	4	2
	g_3	6	5	5

Tabela 3.1: Níveis mínimos de qualificação.

Uma vez que a instância considerada possui partidas que requerem até três árbitros, e como um árbitro não pode atuar em mais de uma PA simultaneamente, serão necessários, no mínimo, três árbitros para completarem todas as PA's de g_1, g_2, e g_3. Além disso, as partidas na localidade a estão associadas a três funções nas equipes de arbitragem: árbitro principal, primeiro auxiliar e segundo auxiliar. Supondo-se que três árbitros sejam suficientes para preencher todas as PA's da referida instância, pode-se determinar o nível mínimo de qualificação para cada um destes árbitros procurando-se pelo maior valor dentre os níveis mínimos de qualificação das PA's associadas a cada função. No exemplo, um árbitro, que seria o principal em todas as partidas, deveria ter um nível de qualificação maior ou igual a 6. Ambos auxiliares deveriam ter níveis de qualificação maiores ou iguais a 5, também para poderem atuar em todas as partidas.

Define-se então $c^1(f, \ell)$ como uma estimativa do número mínimo de árbitros com nível de qualificação ℓ necessários para atuar na localidade f, calculada como descrito no exemplo acima. Neste caso, $c^1(a, 6) = 1$ indica que a localidade a necessita de no mínimo um árbitro com nível de qualificação igual a seis. Da mesma forma, $c^1(a, 5) = 2$ indica que esta localidade necessita de no mínimo dois árbitros com nível de qualificação igual a cinco. Completando, $c^1(a, 1) = c^1(a, 2) = c^1(a, 3) = c^1(a, 4) = 0$.

Seja $c^2(f, \ell)$ o número de PA's sem árbitro atribuído, associados a partidas que ocorrem em f, cujos níveis mínimos de qualificação exigidos são menores ou iguais a ℓ. A Tabela 3.2 mostra os valores de $c^2(a, \ell), \ell = 1, \ldots, 6$, para o mesmo exemplo.

Nível	PA's vazias na localidade a
≤ 1	$c^2(a, 1) = 0$
≤ 2	$c^2(a, 2) = 1 + 0 = 1$
≤ 3	$c^2(a, 3) = 1 + 1 = 2$
≤ 4	$c^2(a, 4) = 2 + 2 = 4$
≤ 5	$c^2(a, 5) = 3 + 4 = 7$
≤ 6	$c^2(a, 6) = 1 + 7 = 8$

Tabela 3.2: Cálculo de c^2.

Assim, para um dado valor de $\overline{p}$, calculado na linha 16 do Pseudo-código 2, o critério guloso, mencionado na linha 17, determina a localidade com a maior necessidade por árbitros com nível de qualificação igual a $\overline{p}$ de forma a maximizar lexicograficamente os seguintes critérios:

- Critério 1: $c^1(f, \overline{p})$ (representa uma estimativa do número mínimo de árbitros com nível de qualificação igual a $\overline{p}$ necessários para atuar na localidade f).
- Critério 2: $c^2(f, \overline{p}) - c^2(f, \overline{p} - 1)$ (consiste no número de PA's sem árbitro atribuído com nível mínimo de qualificação igual a $\overline{p}$ na localidade f, para $\overline{p} > 1$).
- Critério 3: $c^2(f, \overline{p})$ (representa o número de PA's sem árbitro atribuído com nível mínimo de qualificação menor ou igual a $\overline{p}$ na localidade f).
- Critério 4: $c^2(f, MNQ)$ (onde MNQ denota o máximo nível mínimo de qualificação no domínio do problema. Representa o número total de PA's sem árbitro atribuído na localidade f).

Os valores de $c^1(f, \ell)$ e $c^2(f, \ell)$, para todo par (f, ℓ), são atualizados sempre após o final das operações de atribuição do laço das linhas 21 a 29, uma vez que apenas as PA's ainda sem árbitro atribuído são consideradas.

3.1.2
Heurísticas construtivas que priorizam a função objetivo

A heurística construtiva apresentada na seção anterior foi desenvolvida visando a construção de soluções viáveis, sem preocupação com seu valor. Soluções construídas por aquela heurística tendem a ter árbitros atribuídos a partidas em excesso, enquanto outros são designados a um número reduzido ou até mesmo a nenhuma partida.

Com o objetivo de obter soluções iniciais melhores do que aquelas construídas pela heurística anterior, propõem-se dois novos algoritmos com estruturas semelhantes à do primeiro. Os novos algoritmos buscam encontrar soluções nas quais o número de partidas atribuídas a cada árbitro seja mais próximo do seu número desejado de partidas, mesmo que isto implique na violação de algumas restrições.

3.1.2.1
Algoritmo construtivo `CEO_I`

Nesta nova versão, denominada `CEO_I` (Construtivo com Ênfase no Objetivo), uma fase inicial de atribuições limita o número de PA's designadas a cada árbitro ao seu número alvo de partidas. Apenas na segunda fase, após todos os árbitros terem sido considerados e caso ainda existam PA's sem árbitro atribuído, a heurística atribui PA's adicionais para cada árbitro, sempre respeitando o número máximo de partidas de cada um deles.

Assim, o algoritmo `CEO_I` apresenta estrutura muito semelhante àquela exibida no Pseudo-código 2, exceto pelo fato de que cada um dos laços nas linhas 5 a 14 e 15 a 30 são decompostos em duas fases de atribuições. Na primeira, limita-se o número de PA's atribuídas a cada árbitro ao seu número desejado de partidas. Após esta fase inicial, que considera todos os árbitros, caso ainda existam PA's sem árbitro atribuído, executam-se novamente ambos os laços limitando-se agora o número de PA's atribuídas a cada árbitro ao seu número máximo de partidas.

3.1.2.2
Algoritmo construtivo `CEO_II`

Os laços das linhas 9 a 14 e 22 a 26, exibidos no Pseudo-código 2 da heurística `CEV` e presentes também na heurística `CEO_I`, têm como objetivo percorrer as PA's associadas a partidas que ocorrem em uma localidade específica e ainda sem árbitro atribuído. Cada PA é analisada e a viabilidade de sua atribuição ao árbitro em questão é verificada. Claramente, a ordem em que as PA's são

analisadas influencia diretamente na escolha das PA's a serem atribuídas ao árbitro em questão.

Uma terceira heurística construtiva é proposta com base na idéia utilizada no modelo de programação inteira apresentado na Seção 2.5.2, no qual o conjunto de PA's atribuídas a um árbitro é visto como uma sequência em que cada elemento representa uma PA. Esta nova heurística tem o intuito de eliminar a relação entre a solução construída e a ordem em que as PA's são analisadas. Em substituição aos laços das linhas 9 a 13 e 22 a 26 (Pseudo-código 2), computa-se antecipadamente todas as sequências viáveis de PA's para o árbitro em questão e seleciona-se aquela que contiver o número de PA's mais próximo do número desejado pelo árbitro. Esse algoritmo modificado foi denominado `CEO_II` e, à exceção dessa alteração, apresenta a mesma estrutura do algoritmo `CEO_I`. Outra vantagem do algoritmo `CEO_II` com relação aos demais é a possibilidade de consideração de outras funções objetivo, uma vez que as seqüências de PA's para cada árbitro são computadas antecipadamente.

3.2 Esquema baseado na metaheurística ILS

Ambas as heurísticas reparadora e aprimorante, referenciadas no Pseudo-código 1, baseiam-se em esquemas similares da metaheurística ILS (50, 52, 53). Elas se iniciam pela aplicação de um procedimento de busca local à solução inicial. Uma vez que a busca local utilizada envolve movimentos que alteram apenas as atribuições de árbitros que atuam na mesma localidade, esse procedimento é aplicado para cada localidade separadamente.

```
1  Pseudo-código Esquema_ILS(Solucao, MaxIter, MaxIterBL)
2  foreach localidade f do
3      Solucao ← BuscaLocal(f, Solucao, MaxIterBL);
4  end
5  for iter = 1 to MaxIter do
6      NovaSolucao ← Perturbacao(Solucao);
7      Sejam f1 e f2 as localidades envolvidas na perturbação;
8      NovaSolucao ← BuscaLocal(f1, NovaSolucao, MaxIterBL);
9      NovaSolucao ← BuscaLocal(f2, NovaSolucao, MaxIterBL);
10     Solucao ← CriterioDeAceitacao(Solucao, NovaSolucao);
11 end
12 return Solucao;
```

Pseudo-código 3: Esquema baseado na metaheurística ILS.

Em seguida, por um certo número de iterações, submete-se a solução corrente a uma perturbação, envolvendo um par de localidades, que visa mudar

a localidade onde atua pelo menos um árbitro. Cada perturbação é sucedida por duas aplicações do procedimento de busca local, uma para cada localidade envolvida na referida perturbação. A solução obtida após este processo é aceita caso satisfaça um certo critério de aceitação. Este esquema é ilustrado pelo Pseudo-código 3.

A próxima seção descreve o procedimento de busca local e as respectivas vizinhanças. Em seguida, são apresentados detalhes específicos das heurísticas reparadora e aprimorante para o PAA básico.

3.2.1 Busca local e vizinhanças

As soluções construídas pelas heurísticas descritas nas seções anteriores não são necessariamente ótimas ou mesmo viáveis. Um algoritmo de busca local substitui sucessivamente a solução atual por uma melhor, que esteja em sua vizinhança, até que atinja um ótimo local, ou seja, uma solução que não possua nenhum vizinho de menor custo que ela própria. Na estratégia *primeiro aprimorante*, o primeiro vizinho encontrado cujo custo seja menor que o da solução atual é escolhido para substituí-la. Neste trabalho, duas vizinhanças são consideradas:

- movimentos de dupla-troca: os árbitros atribuídos a duas PA's são trocados entre si (tais movimentos não alteram o número de partidas atribuídas a cada árbitro) e
- movimentos de simples-troca: o árbitro atribuído a uma PA é substituído por outro (tais movimentos incrementam em uma unidade o número de partidas a que um dos árbitros está designado e decrementa também em uma unidade o número de partidas em que o outro árbitro atua).

Como os árbitros não podem ser designados a partidas que acontecem em localidades diferentes (restrição forte), apenas movimentos envolvendo árbitros que atuam na mesma localidade (ou não arbitram partida alguma) são permitidos (de outra forma, a menos que os dois árbitros estivessem atribuídos a exatamente uma partida cada, um movimento envolvendo árbitros que atuam em diferentes localidades implicaria em pelo menos uma restrição violada). Tais vizinhanças restritas que consideram movimentos em cada localidade separadamente permitem uma aceleração do procedimento de busca local.

A busca local realizada dentro do esquema baseado em ILS é dividida em quatro fases e está ilustrada no Pseudo-código 4. Na primeira fase (`primeiroAprimorante_ST`), um procedimento de busca local tradicional é aplicado utilizando a estratégia *primeiro aprimorante*. Apenas movimentos de

simples-troca são considerados e somente movimentos aprimorantes são aceitos. A segunda fase (`primeiroAprimorante_DT`) consiste também em um procedimento de busca local tradicional, considerando agora movimentos aprimorantes de dupla-troca. Uma vez que tais movimentos não alteram o valor da função objetivo considerada até o momento, esta fase não existe no caso da heurística aprimorante para o PAA básico. Na terceira fase (`tabu_ST`), consideram-se novamente apenas os movimentos de simples-troca. Entretanto, agora são aceitos movimentos que mantenham ou reduzam o custo da solução atual, utilizando-se uma lista de movimentos proibidos para prevenir a ocorrência de ciclos. Finalmente, na quarta fase (`tabu_DT`), os movimentos de dupla-troca passam a ser considerados e são aceitos movimentos que pelo menos mantenham o custo da solução atual. Novamente utiliza-se uma lista de movimentos proibidos para prevenir a ocorrência de ciclos. A execução destas fases é repetida por um certo número `MaxIterBL` de iterações.

```
1  Pseudo-código BuscaLocal(f, Solucao, MaxIterBL)
2  for iter = 1 to MaxIterBL do
3      Solucao ← primeiroAprimorante_ST(f, Solucao);
4      SolucaoBackUp ← Solucao;
5      Solucao ← primeiroAprimorante_DT(f, Solucao);
6      if SolucaoBackUp = Solucao then
7          Solucao ← tabu_ST(f, Solucao);
8          if SolucaoBackUp = Solucao then
9              Solucao ← tabu_DT(f, Solucao);
10             if SolucaoBackUp = Solucao then
11                 return Solucao;
12 end
13 return Solucao;
```

Pseudo-código 4: Procedimento de busca local.

3.2.2 Busca local como um problema de programação linear inteira

O procedimento de busca local descrito na Seção 3.2.1 é utilizado no algoritmo baseado na metaheurística ILS a cada iteração, sempre após uma perturbação. Dada uma localidade f envolvida na referida perturbação, os movimentos aplicados à solução atual alteram atribuições envolvendo árbitros e PA's que se encontram nesta localidade. Esta busca local resolve uma instância reduzida do PAA, procurando otimizar as atribuições dos árbitros que atuam na localidade f. Nesta seção, propõe-se uma abordagem híbrida, sugerida por Fischetti e Lodi (29), que substitui o procedimento de busca local aplicado após

cada perturbação pela resolução exata do modelo de programação linear inteira (3-1)-(3-7) abaixo, associado às localidades f_1 e f_2 envolvidas nas perturbações realizadas no Pseudo-código 3.

Sejam $P(f) \subseteq S$ o conjunto de índices de PA's associadas às partidas que acontecem na localidade f e $Q(f) \subseteq R$ o conjunto de índices de árbitros designados a partidas na localidade f, na solução atual.

O sub-problema associado a cada par de localidades pode então ser formulado como:

$$\text{minimizar} \sum_{i \in Q(f_1) \cup Q(f_2)} d_i \tag{3-1}$$

sujeito a:

$$d_i = |T_i - \sum_{j \in P(f_1) \cup P(f_2)} x_{ij}| \quad \forall i \in Q(f_1) \cup Q(f_2) \tag{3-2}$$

$$\sum_{i \in Q(f_1) \cup Q(f_2)} x_{ij} = 1 \quad \forall j \in P(f_1) \cup P(f_2) \tag{3-3}$$

$$\sum_{j \in P(f_1) \cup P(f_2)} x_{ij} \leq M_i \quad \forall i \in Q(f_1) \cup Q(f_2) \tag{3-4}$$

$$x_{ij} + x_{ij'} \leq 1 \quad \forall i \in Q(f_1) \cup Q(f_2), \quad \forall j \in P(f_1) \cup P(f_2), \quad \forall j' \in C(j) \tag{3-5}$$

$$\sum_{j \in U(i)} x_{ij} = 0 \quad \forall i \in Q(f_1) \cup Q(f_2) \tag{3-6}$$

$$x_{ij} \in \{0,1\} \quad \forall i \in Q(f_1) \cup Q(f_2), \quad \forall j \in P(f_1) \cup Q(f_2), \tag{3-7}$$

onde

$$x_{ij} = \begin{cases} 1, \text{ se o árbitro } i \in Q(f_1) \cup Q(f_2) \text{ está atribuído à PA } j \in P(f_1) \cup P(f_2) \\ 0, \text{ caso contrário.} \end{cases}$$

O modelo (3-1) a (3-7) corresponde ao modelo (2-1) a (2-7) da Seção 2.5.1, quando são fixadas todas as atribuições da solução corrente e liberadas somente aquelas dos árbitros atuando em partidas que acontecem nas localidades f_1 e f_2.

A função objetivo (3-1) determina que seja minimizada a soma sobre todos os árbitros que atuam nas localidades f_1 ou f_2 do valor absoluto da diferença entre os números de partidas arbitradas e desejadas. As restrições (3-2) estabelecem o valor de d_i, para $i \in Q(f_1) \cup Q(f_2)$. As restrições (3-3) garantem que toda PA associada a partidas que ocorrem nas localidades f_1 ou f_2 deve ser atribuída a exatamente um árbitro. As restrições (3-4) estabelecem um limite superior para o número de PA's que podem ser atribuídas a cada árbitro. As restrições (3-5) impedem que PA's associadas a partidas em conflitos de horários sejam atribuídas a um mesmo árbitro. As restrições (3-6) impedem atribuições que violem os requisitos mínimos de qualificação e

disponibilidades. As restrições (3-7) estabelecem a integralidade das variáveis de decisão.

3.2.3 Heurística reparadora

A heurística reparadora proposta, referenciada na linha 4 do Pseudo-código 1, se baseia no esquema apresentado pelo Pseudo-código 3, que consiste basicamente na aplicação iterativa de busca local e perturbações. Seu objetivo é tornar viável uma solução inviável inicial obtida pela heurística construtiva. As violações de restrições na solução inicial podem consistir em conflitos de horários, em indisponibilidades dos árbitros não respeitadas, em níveis de qualificação inadequados ou em número excessivo de partidas atribuídas a alguns árbitros. A heurística reparadora minimiza o número de restrições violadas na solução inicial. Uma solução é considerada como viável se e somente se não apresentar restrição violada alguma.

Todos os árbitros atuam em no máximo uma localidade nas soluções produzidas pelas heurísticas construtivas. Assim, a busca local considera apenas movimentos envolvendo árbitros que atuam numa mesma localidade (ou não atuam em partida alguma). No caso de duas soluções apresentarem o mesmo número de violações, é escolhida aquela com menor valor da função objetivo.

O procedimento de perturbação utilizado pela heurística reparadora altera a localidade onde atua um dos árbitros, de acordo com os seguintes passos:

1. Selecionar aleatoriamente uma localidade f com pelo menos uma atribuição inviável.

2. Determinar o maior nível mínimo de qualificação p^* dentre aqueles exigidos pelas PA's na localidade f e atribuídas a árbitros que possuem pelo menos uma restrição violada.

3. Selecionar aleatoriamente um árbitro i que atua em outra localidade f' (ou que não atua em partida alguma) e cujo nível de qualificação seja maior ou igual a p^*.

4. Escolher aleatoriamente outros árbitros diferentes de i que já atuem na localidade f' e atribuir a tais árbitros as PA's que atualmente estão designadas a i.

5. Atribuir ao árbitro i uma PA escolhida aleatoriamente dentre aquelas associadas a partidas que ocorrem na localidade f e que atualmente encontram-se atribuídas a árbitros com pelo menos uma violação.

A solução `NovaSolucao` obtida após uma perturbação sucedida por busca local é aceita pelo procedimento `CriterioDeAceitacao`, no Pseudo-código 3, se e somente se ela possui menos violações (ou o mesmo número de violações e um valor menor para a função objetivo) do que a solução corrente. O procedimento é repetido até que uma solução viável seja encontrada ou que um certo critério de parada seja satisfeito.

3.2.4 Heurística aprimorante

Uma vez que uma solução viável tenha sido obtida, a heurística aprimorante, referenciada na linha 6 do Pseudo-código 1, é executada para melhorá-la. Esta heurística, assim como a reparadora, se baseia no esquema apresentado na Seção 3.2 baseado na metaheurística ILS.

A busca local utilizada na heurística aprimorante difere ligeiramente daquela usada na heurística reparadora. Movimentos de dupla-troca não são realizados na fase que busca somente por soluções aprimorantes, pois tais movimentos não alteram o valor da função objetivo. Outra diferença é que apenas movimentos que preservam a viabilidade da solução atual são considerados.

As perturbações aplicadas pela heurística aprimorante selecionam dois árbitros que atuam em diferentes localidades e realizam a troca de todas suas atribuições entre si, de acordo com os passos exibidos no Pseudo-código 5. O primeiro par de árbitros encontrado que resulte em uma perturbação classificada como promissora tem suas atribuições trocadas e a solução resultante desta troca é retornada. O conceito adotado para determinar se uma perturbação é promissora ou não será discutido adiante. Caso nenhuma perturbação promissora seja encontrada, o procedimento retorna a solução resultante da perturbação que menos aumente o valor da solução corrente.

A solução $\mathtt{S}^*$ que representa a melhor solução não promissora já obtida pela perturbação, é inicializada na linha 2. Entende-se que uma solução é não promissora se ela foi obtida por uma perturbação não promissora. O laço das linhas 3 a 12 percorre todos os pares de árbitros disponíveis. Na linha 4, determinam-se as localidades para as quais estão designados os árbitros selecionados. Na linha 5, verifica-se se estes árbitros estão designados a localidades diferentes. Em caso afirmativo, na linha 6 obtém-se a solução $\mathtt{S}'$ pela troca das atribuições desses árbitros. Se a solução $\mathtt{S}'$ é viável (teste da

```
1  Pseudo-código Perturbacao_ILS_Aprimorante(S)
2  S* ← ∅;
3  foreach (i, i') ∈ R × R do
4      Sejam f1 e f2 as localidades onde arbitram respectivamente i e i'
       na solução S;
5      if f1 ≠ f2 then
6          Seja S' a solução obtida pela troca das atribuições entre i e i';
7          if S' é viável then
8              if Promissora(S', i, i') then
9                  return S';
10             if S* = ∅ ou o valor objetivo de S* é maior que o de S'
               then
11                 S* ← S';
12 end
13 return S*;
```

Pseudo-código 5: Esquema da perturbação realizada na heurística aprimorante.

linha 7), na linha 8 verifica-se se esta perturbação é considerada promissora. Em caso afirmativo, esta solução é retornada na linha 9 e o procedimento encerrado. Caso contrário, nas linhas 10 e 11 atualiza-se a melhor solução não promissora ($\mathtt{S}^*$) encontrada até o momento. Ao final, caso nenhuma perturbação promissora tenha sido encontrada, a solução $\mathtt{S}^*$ é retornada na linha 13.

O procedimento $\mathtt{Promissora}(\mathtt{S}', i, i')$, executado no teste da linha 8, retorna verdadeiro se o valor absoluto da diferença entre o número desejado de partidas e o número de partidas efetivamente arbitradas por i ou por i' pode ser trivialmente reduzido após a troca, retornando falso caso contrário. Este procedimento consiste em um algoritmo de *look ahead* que tem como objetivo verificar se a troca dos árbitros i e i' é promissora ou não. Se o número de partidas atribuídas a cada um destes árbitros já for exatamente igual ao seu número alvo de partidas, esta perturbação não é considerada promissora, pois o valor objetivo associado a cada um desses árbitros não pode ser reduzido. Por outro lado, se o número desejado de partidas de algum dos árbitros for superior ao número de partidas arbitradas por ele após a troca para a nova localidade, procura-se por novas PA's, associadas a partidas que ocorrem nesta nova localidade, que possam ser atribuídas a este árbitro, de forma que ele seja designado a um número de partidas mais próximo de seu número alvo do que no momento após a troca de localidade. Somente são avaliados nesse procedimento as PA's que estejam atribuídas a árbitros que tenham sido designados a um número de partidas maior que seu número alvo (caso contrário, uma troca

de atribuição resultaria na diminuição do valor da função objetivo de um dos árbitros e no aumento do valor da função objetivo do outro).

A solução obtida após a perturbação e sucedida por uma busca local é sempre aceita, pois o critério utilizado na escolha da perturbação já seleciona a primeira perturbação promissora (ou, no caso de nenhuma ser promissora, aquela que menos incremente o custo da solução corrente). Assim como a heurística reparadora, este procedimento é executado até que o critério de parada escolhido seja satisfeito.

3.3 Heurística de três fases

A variante básica da heurística de três fases que combina um algoritmo construtivo, a heurística reparadora e a heurística aprimorante, realizando as perturbações da fase aprimorante como descrito na seção anterior, foi denominada simplesmente de `3fases`. No caso em que o algoritmo construtivo escolhido é `CEV`, tem-se a heurística denominada `3fases + CEV` nos experimentos computacionais apresentados a seguir. Um estudo computacional para esta variante foi apresentado originalmente em (13).

O procedimento de *look ahead*, utilizado na escolha das perturbações, se fez necessário para melhorar os resultados obtidos por perturbações selecionadas aleatoriamente. No entanto, tal procedimento pode ter que avaliar um grande número de pares de árbitros, a cada iteração. Em instâncias com um grande número de árbitros, este procedimento de busca exaustiva pode se mostrar bastante ineficiente. Com o objetivo de avaliar a necessidade ou não de explorar todas as possibilidades de trocas de árbitros (combinações dois a dois de árbitros que atuam em diferentes localidades), foi proposta uma variante `3fases_II` que substitui a avaliação completa da vizinhança por uma busca que avalia um número limitado das combinações possíveis.

Adicionalmente, a heurística `3fases_II` apresenta uma modificação no procedimento de *look ahead*. Uma vez que, da forma descrita acima, são verificadas possíveis melhorias somente para os árbitros que, após a troca para a nova localidade, sejam designados a um número menor de partidas que seu número alvo. Nesta nova variante, avalia-se também a possibilidade de melhoria para árbitros designados a um número de partidas maior que seu número alvo. Procura-se então por outros árbitros, que já estejam arbitrando partidas nessa nova localidade e que possam ser designados a alguma partida atribuída ao árbitro avaliado. Considera-se na busca apenas os árbitros que atualmente estejam atribuídos um número de partidas menor que seu alvo, pois caso contrário uma troca não resultaria na melhora do valor da função

objetivo.

3.4 Resultados computacionais

Os resultados experimentais relatados nesta seção foram obtidos com um processador AMD Athlon^{TM}1550 MHz, com 512 MBytes de memória RAM e sistema operacional Windows 2000^{TM}. Todos os algoritmos foram implementados em linguagem C. A geração de números aleatórios se baseou no procedimento proposto em (68). O resolvedor de programação inteira utilizado foi o CPLEX^{TM}versão 8.0. O único experimento que não foi realizado com a plataforma descrita acima foi o que compara os resultados obtidos pela resolução do modelo de programação inteira com os resultados da heurística de três fases. Uma vez que a resolução exata, descrita no capítulo anterior, se deu em outra plataforma, ela foi também utilizada neste experimento para a abordagem heurística descrita neste capítulo.

As instâncias utilizadas nos experimentos foram construídas artificialmente como descrito na Seção 2.6.1. Para ilustrar o comportamento dos algoritmos aproximados, foram selecionados três grupos de instâncias com 500 partidas e 750 árbitros, diferentes números de localidades (65 e 85) e diferentes padrões para geração dos números desejados de partidas para cada árbitro. Estas instâncias foram escolhidas por seus resultados serem bem representativos daqueles observados para o grupo completo de instâncias.

3.4.1 Algoritmos construtivos

O objetivo do primeiro experimento foi permitir uma comparação entre os métodos usados para a construção de soluções. Os resultados numéricos foram obtidos pelas fases de construção e reparação da heurística de três fases para as instâncias mencionadas acima. O critério de parada para a heurística reparadora foi a execução de 10000 iterações.

As Tabelas 3.3 a 3.5 exibem os resultados referentes a este primeiro experimento realizado e foram obtidos com dez execuções para cada instância, com diferentes sementes para a geração de números aleatórios. A primeira coluna indica a instância utilizada. As quatro colunas seguintes trazem informações associadas à fase de construção, respectivamente a identificação do método utilizado, o tempo gasto em segundos, o valor objetivo da solução viável encontrada e o número de soluções viáveis encontradas nas dez execuções. Os tempos de processamento e os valores da função objetivo são resultados médios para as dez execuções para cada instância. O valor médio do objetivo foi

assinalado com "-" nos casos em que a heurística construtiva não encontrou solução viável alguma. As últimas três colunas das tabelas exibem as informações relativas à heurística reparadora, respectivamente o tempo gasto em segundos, o valor médio do objetivo das soluções viáveis encontradas e o número de soluções viáveis encontradas nas execuções em que esta fase foi necessária, ou seja, quando o algoritmo construtivo não encontrou solução viável. Novamente, os tempos de processamento e os valores da função objetivo são resultados médios. Nos casos em que a heurística construtiva encontrou soluções viáveis em todas as dez execuções, a fase de reparação não foi necessária e as respectivas colunas nas tabelas aparecem assinaladas com "-".

Instância	Construção				Reparação		
id.	método	tempo (s)	valor	viáveis	tempo (s)	valor	viáveis
I_1	RAND	0,01	-	0	8,03	976,20	10
	CEV	0,02	1273,40	10	-	-	-
	CEO_I	0,07	-	0	0,89	713,20	10
	CEO_II	0,11	-	0	2,05	801,60	10
I_2	RAND	0,01	-	0	7,71	1016,00	10
	CEV	0,02	1354,60	10	-	-	-
	CEO_I	0,07	-	0	1,04	771,60	10
	CEO_II	0,20	-	0	2,43	853,20	10
I_3	RAND	0,01	-	0	7,47	1033,60	10
	CEV	0,02	1353,60	10	-	-	-
	CEO_I	0,07	-	0	0,26	726,80	10
	CEO_II	0,13	-	0	1,81	848,40	10
I_4	RAND	0,01	-	0	9,35	947,30	10
	CEV	0,02	1313,00	10	-	-	-
	CEO_I	0,06	-	0	0,76	754,40	10
	CEO_II	0,08	-	0	1,76	836,40	10
I_5	RAND	0,01	-	0	11,32	960,60	10
	CEV	0,01	1264,40	10	-	-	-
	CEO_I	0,06	-	0	1,17	755,80	10
	CEO_II	0,10	-	0	1,95	827,80	10

Tabela 3.3: Instâncias com 65 localidades e padrão P_0.

Além das três heurísticas construtivas propostas na Seção 3.1 (CEV, CEO_I e CEO_II), foi incluída no experimento uma variante indicada por RAND, na qual as soluções foram construídas de forma completamente aleatória. Esta variante foi importante para a avaliação do comportamento da heurística reparadora.

As heurísticas construtivas se mostraram bastante eficientes, não ultrapassando $0,20$ segundo para instância alguma nas Tabelas 3.3 a 3.5. A heurística CEV encontrou soluções viáveis na maioria dos casos (108 soluções viáveis em 150 execuções). Este algoritmo teve maior dificuldade em encontrar

Instância		Construção			Reparação		
id.	método	tempo (s)	valor	viáveis	tempo (s)	valor	viáveis
I_1	RAND	0,01	-	0	13,62	1326,40	10
	CEV	0,02	1751,80	10	-	-	-
	CEO_I	0,05	-	0	1,88	1079,00	10
	CEO_II	0,11	-	0	2,91	1102,80	10
I_2	RAND	0,01	-	0	19,54	1318,00	10
	CEV	0,02	1788,20	10	-	-	-
	CEO_I	0,05	-	0	1,92	1119,00	10
	CEO_II	0,09	-	0	3,86	1148,20	10
I_3	RAND	0,01	-	0	7,44	1327,80	10
	CEV	0,02	1894,80	10	-	-	-
	CEO_I	0,05	1007,14	7	0,11	1017,33	3
	CEO_II	0,12	-	0	1,11	1054,80	10
I_4	RAND	0,01	-	0	11,76	1237,20	10
	CEV	0,02	1746,80	10	-	-	-
	CEO_I	0,05	-	0	1,02	1005,00	10
	CEO_II	0,08	-	0	2,63	1030,80	10
I_5	RAND	0,01	-	0	10,41	1266,60	10
	CEV	0,02	1761,40	10	-	-	-
	CEO_I	0,05	-	0	1,18	1053,20	10
	CEO_II	0,10	-	0	2,70	1079,00	10

Tabela 3.4: Instâncias com 65 localidades e padrão P_1.

soluções viáveis somente para as instâncias com 85 localidades (apenas oito soluções viáveis em 50 execuções). Porém, mesmo assim as soluções inviáveis construídas por esta heurística foram aquelas que exigiram os menores tempos pela heurística reparadora, como pode ser observado na Tabela 3.5. As heurísticas CEO_I e CEO_II, apesar de não terem sido tão eficientes quanto o algoritmo CEV, no que se refere a encontrar soluções viáveis (por exemplo, na Tabela 3.4, a heurística CEO_I encontrou apenas sete soluções viáveis para a instância I_3 com 65 localidades e padrão P_1 e a heurística CEO_II não encontrou solução viável alguma), mostraram que obtêm soluções que exigem baixos tempos de processamento da fase de reparação. Comparando-se com as soluções construídas de forma completamente aleatória, as três heurísticas construtivas obtiveram soluções que exigem tempos de processamento muito menores da fase de reparação, mostrando que mesmo priorizando o objetivo os algoritmos CEO_I e CEO_2 não deixam de considerar a viabilidade da solução construída.

Por sua vez, a heurística reparadora obteve soluções viáveis em todos os casos em que as heurísticas construtivas falharam. Este algoritmo também obteve soluções viáveis na maioria dos casos mesmo quando partiu de soluções

geradas aleatoriamente (falhou apenas em 22 de um total de 150 execuções, apenas em instâncias com 85 localidades). Observa-se que as instâncias com maior número de localidades (para números fixos de partidas e árbitros) ofereceram maior dificuldade na busca por soluções viáveis (exigindo maiores tempos de processamento da heurística reparadora). A heurística reparadora falhou em alguns casos em que partiu de soluções aleatórias, entretanto obteve sucesso em todos os casos em que uma heurística construtiva foi utilizada. Esta comparação reforça a importância da utilização de procedimentos eficientes para encontrar soluções viáveis iniciais para problemas combinatórios em esportes, como já havia sido observado por Ribeiro e Urrutia (65).

Instância		Construção			Reparação		
id.	método	tempo (s)	valor	viáveis	tempo (s)	valor	viáveis
I_1	RAND	0,01	-	0	79,56	1056,29	7
	CEV	0,03	-	0	12,01	1158,60	10
	CEO_I	0,07	-	0	19,35	942,80	10
	CEO_II	0,07	-	0	26,74	976,20	10
I_2	RAND	0,01	-	0	102,82	1062,00	3
	CEV	0,02	-	0	10,35	1136,00	10
	CEO_I	0,02	-	0	25,95	934,40	10
	CEO_II	0,09	-	0	32,32	986,60	10
I_3	RAND	0,01	-	0	48,31	1088,89	9
	CEV	0,02	1266,25	8	0,76	1262,00	2
	CEO_I	0,07	-	0	9,73	970,60	10
	CEO_II	0,07	-	0	14,00	1024,40	10
I_4	RAND	0,01	-	0	97,06	1027,50	4
	CEV	0,03	-	0	4,73	1167,00	10
	CEO_I	0,06	-	0	26,06	945,60	10
	CEO_II	0,06	-	0	25,42	968,60	10
I_5	RAND	0,01	-	0	96,42	1018,60	5
	CEV	0,02	-	0	5,68	1172,20	10
	CEO_I	0,02	-	0	25,07	907,80	10
	CEO_II	0,09	-	0	36,22	953,00	10

Tabela 3.5: Instâncias com 85 localidades e padrão P_0.

Um segundo experimento teve o intuito de comparar os resultados apresentados para as variantes da heurística de três fases baseada em busca local por vizinhança combinadas aos diferentes algoritmos construtivos (inicialmente não é considerada a busca local por programação inteira, avaliada posteriormente em outro experimento). Consideram-se então as seguintes variantes:

– A heurística original com o algoritmo construtivo `CEV` (`3fases + CEV`)

– A heurística otimizada com o algoritmo construtivo `CEV` (`3fases_II + CEV`)

– A heurística otimizada com o algoritmo construtivo `CEO_I` (`3fases_II + CEO_I`)

– A heurística otimizada com o algoritmo construtivo `CEO_II` (`3fases_II + CEO_II`)

A heurística otimizada (`3fases_II`) foi configurada para estes experimentos de forma a avaliar somente 10% das combinações possíveis de pares de árbitros no procedimento de escolha das perturbações da heurística aprimorante.

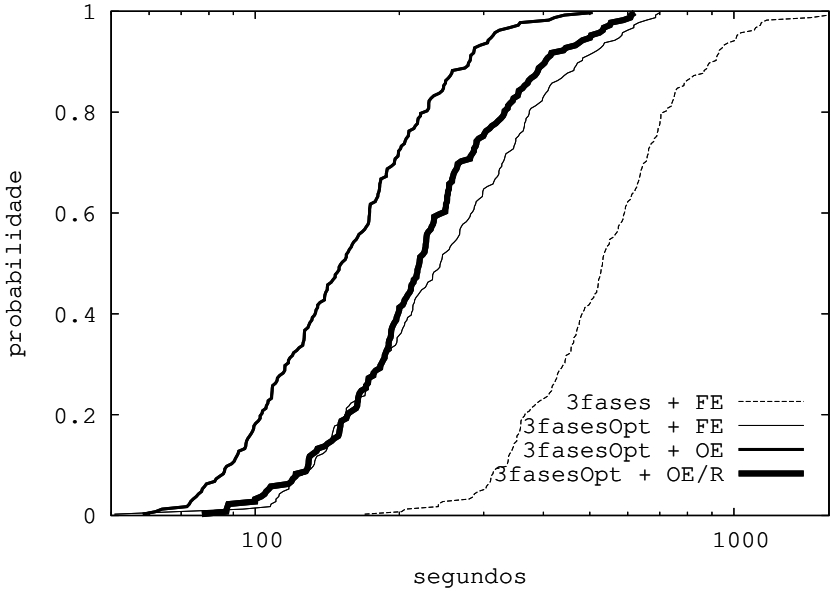

Figura 3.1: Instância I_1 com 65 localidades e padrão P_0 (alvo: 364)

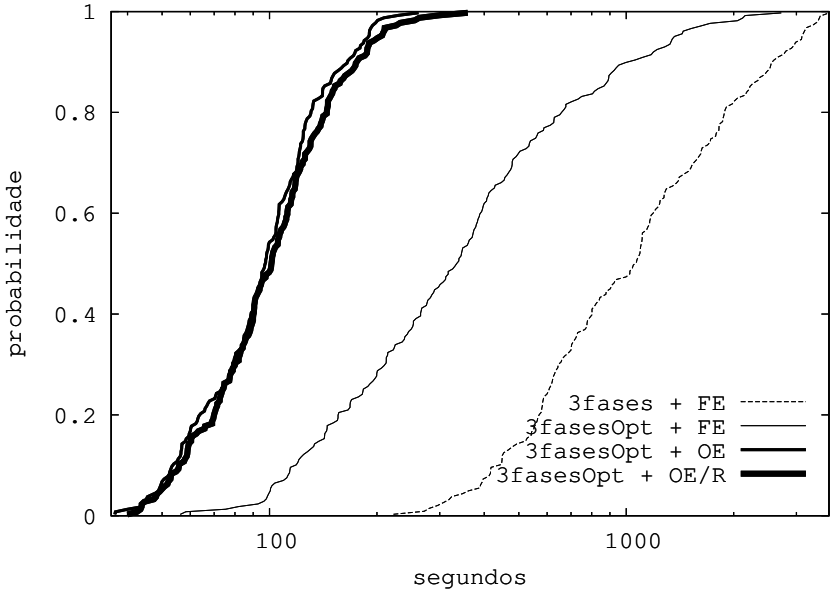

Figura 3.2: Instância I_1 com 65 localidades e padrão P_1 (alvo: 837)

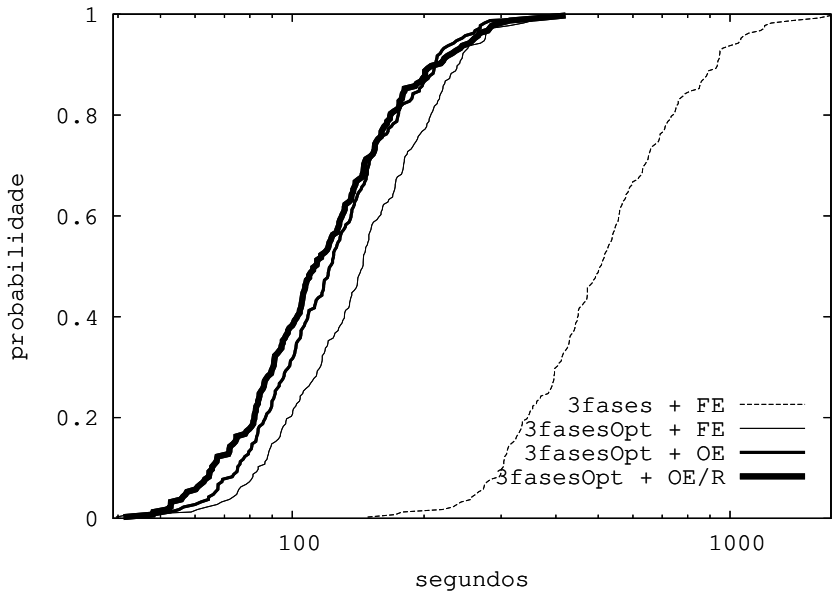

Figura 3.3: Instância I_1 com 85 localidades e padrão P_0 (alvo: 514)

A comparação das diferentes variantes é baseada nos tempos de processamento necessários para que cada uma encontre uma solução pelo menos tão boa quanto um certo valor alvo determinado a priori. Nesta comparação foram utilizados gráficos denominados TTT (do inglês *time-to-target*). Segundo Aiex, Resende, e Ribeiro (1), estes gráficos mostram distribuições empíricas da variável aleatória *time-to-target*, que é o tempo gasto para atingir uma solução alvo pelos diferentes algoritmos, para determinado par de instância e valor alvo. Para se construir a distribuição empírica, escolhe-se um valor alvo e executa-se cada algoritmo 200 vezes, armazenando-se o tempo gasto para encontrar a primeira solução com custo menor ou igual ao alvo estipulado. Para cada algoritmo, associa-se o i-ésimo tempo de processamento t_i (tempos ordenados em ordem não decrescente) com a probabilidade $p_i = (i - \frac{1}{2})/200$ e traça-se o gráfico com os pontos $z_i = (t_i, p_i)$, para $i = 1, \ldots, 200$. Observa-se que quanto mais "à esquerda" uma curva aparece em um gráfico TTT, melhor é o algoritmo correspondente (uma vez que leva menos tempo para encontrar o valor alvo com uma dada probabilidade). Os gráficos TTT foram também utilizados por Feo, Resende e Smith (26) e são defendidos por Stützle (41, 42, 43) como uma forma de caracterizar os tempos de processamento de algoritmos estocásticos para otimização combinatória. Tais gráficos também foram usados em diversos estudos computacionais como (1, 2, 7, 27, 34, 59, 60, 61, 66).

As Figuras 3.1 a 3.3 exibem resultados ilustrativos para as quatro variantes dos algoritmos (`3fases + CEV`, `3fases_II + CEV`, `3fases_II + CEO_I` e `3fases_II + CEO_II`) para três instâncias teste selecionadas. Valores alvo difíceis foram selecionados para todas as instâncias, respectivamente 364, 837

e 514. Estes valores foram determinados em experimentos preliminares com a heurística original (`3fases + CEV`) e representam, para cada instância, a média do valor objetivo das soluções encontradas em dez execuções, cada uma com o tempo de execução limitado em 600 segundos.

Os resultados mostram que a versão otimizada da heurística de três fases (`3fases_II + CEV`) é mais eficiente que a versão original (`3fases + CEV`), pois dado um limite de tempo fixo a probabilidade de que ela encontre uma solução pelo menos tão boa quanto o alvo estabelecido é sistematicamente maior que a associada à heurística original. Ou seja, a decisão de avaliar somente parte de uma vizinhança muito grande aliada a um procedimento de *look ahead* mais elaborado levou a heurística de três fases a obter melhores resultados.

As heurísticas construtivas gulosas `CEO_I` e `CEO_II` apresentaram resultados muito semelhantes para duas instâncias (Figuras 3.2 e 3.3), entretanto o algoritmo `CEO_I` foi melhor para a outra instância (Figura 3.1).

Concluindo, os algoritmos que priorizam a função objetivo apresentaram melhores resultados do que aquele que visa somente a viabilidade. Este resultado reforça a idéia de que métodos que constroem boas soluções iniciais são importantes para o sucesso da abordagem de resolução. Comparando-se os algoritmos `CEO_I` e `CEO_II`, o primeiro obteve resultados ligeiramente melhores, mostrando que o método de seleção das atribuições utilizado pelo segundo (que prioriza a melhor sequência de atribuições para cada árbitro), apesar de representar a melhor escolha localmente, não resultou em melhores soluções. Entretanto, mesmo tendo apresentado resultado ligeiramente inferior, o algoritmo construtivo `CEO_II` será usado nos próximos experimentos por permitir a utilização de diferentes funções de custo. Essa flexibilidade possibilitará sua posterior utilização na abordagem bi-objetivo para o PAA, descrita no Capítulo 5.

3.4.2 Função de custo linear vs. quadrática

Em outro experimento, a função objetivo linear foi substituída por uma penalização quadrática para o valor absoluto da diferença entre os números de jogos arbitrados e desejados por cada árbitro. Os resultados deste experimento são ilustrados pela Tabela 3.6, que fornece os resultados para a instância I_3 com 85 localidades e padrão P_0.

Esta tabela detalha as diferenças entre os números de partidas desejadas e arbitradas por cada árbitro nas soluções obtidas com as funções de custos linear e quadrática. Pode-se observar que dos 255 árbitros privilegiados (isto é, aqueles atribuídos ao número exato de partidas desejadas) na solução obtida

$\|D\|$	Número de árbitros	
	linear	quadrática
0	255	179
1	182	281
2	156	149
3	67	66
4	50	43
5	23	18
6	13	10
7	3	3

Tabela 3.6: Comparação entre as funções de custo linear e quadrática. $|D|$ é o valor absoluto da diferença, para cada árbitro, entre os números de partidas desejado e o observado. As colunas do meio e da direta trazem as quantidades de árbitros na solução que apresentam a diferença exibida na coluna da esquerda para as funções objetivo linear e quadrática, respectivamente.

com a função linear, 76 perdem este privilégio na solução obtida pela função quadrática. Por outro lado, o número de árbitros com diferença igual a uma unidade subiu de 182 para 281, fato que permitiu que 23 árbitros que estavam longe do número desejado de partidas (diferenças maiores ou iguais a duas unidades) ficassem mais próximos (diferenças iguais a uma unidade). A nova solução obtida com a função de custos quadrática é certamente mais balanceada do que aquela associada à função linear. Os tempos computacionais gastos pelas heurísticas construtiva, reparadora e aprimorante não foram afetados pela mudança na função objetivo.

Entretanto, para a utilização da abordagem de busca local por programação linear inteira, os demais experimentos apresentados neste trabalho seguirão baseados na função objetivo linear. Todavia, para a abordagem com busca local por vizinhanças a mudança na função objetivo é uma estratégia vantajosa caso haja interesse pelo balanceamento das soluções, uma vez que os tempos de processamento dos algoritmos não são afetados.

3.4.3 Heurística híbrida com modelo de PI

Como descrito na Seção 3.2.2, o procedimento de busca local por vizinhanças realizado em cada iteração da heurística aprimorante pode ser substituído pela resolução exata de um modelo de programação linear inteira para cada sub-problema associado às localidades envolvidas na perturbação. Este experimento teve como intuito comparar os tempos gastos para a resolução dos mencionados sub-problemas, utilizando cada um dos diferentes modelos de programação linear inteira propostos no capítulo anterior.

Instância			Modelo 1	Modelo 2	Modelo 3
$\|F\|$	padrão	id.	tempo (s)	tempo (s)	tempo (s)
65	P_0	I_1	0,06	3,12	2,15
		I_2	0,05	5,32	3,09
		I_3	0,06	6,23	3,50
		I_4	0,05	1,97	1,19
		I_5	0,06	2,54	1,59
65	P_1	I_1	0,05	4,31	2,19
		I_2	0,04	3,63	2,04
		I_3	0,04	3,69	2,19
		I_4	0,04	1,86	1,09
		I_5	0,05	2,82	1,58
85	P_0	I_1	0,04	0,78	0,51
		I_2	0,04	1,33	0,76
		I_3	0,04	0,62	0,42
		I_4	0,04	0,68	0,44
		I_5	0,05	1,34	0,85

Tabela 3.7: Tempos médios da variante `3fases_II + CEV + MIP` para 100 iterações relativos somente à resolução do modelo de programação inteira que substitui o procedimento de busca local após cada perturbação da ILS.

A Tabela 3.7 exibe os tempos médios (em segundos) gastos pelo CPLEX com cada modelo observados sobre um total de 100 iterações da heurística de três fases (`3fases_II + CEV + MIP`). Estes tempos são relativos à resolução exata dos sub-problemas associados às duas localidades envolvidas em cada uma das 100 perturbações.

Ao contrário dos resultados exibidos na comparação entre os modelos na Seção 2.6.2 do capítulo anterior, neste experimento observa-se que os tempos de processamento gastos para a resolução do sub-problema utilizando o Modelo 1 foram muito menores que os tempos gastos usando os outros modelos. Este resultado pode ser explicado pelos tempos gastos (exibidos na Seção 2.6.2) para a resolução da relaxação linear utilizando-se cada modelo. Uma vez que os sub-problemas resolvidos dentro da heurística híbrida são de pequeno porte, a eficiência da resolução da relaxação do Modelo 1 foi mais determinante que os valores mais apertados para os Modelos 2 e 3. Dessa forma, o Modelo 1 foi selecionado para ser utilizado em conjunto com a heurística híbrida nos experimentos subseqüentes.

Escolhido o Modelo 1 para a resolução exata, em substituição ao procedimento de busca local baseada em vizinhanças, as Figuras 3.4 a 3.6 exibem três gráficos TTT para a comparação de quatro variantes da heurística otimizada, duas baseadas em busca local por vizinhanças e outras duas híbridas que substituem a busca local pela resolução de um sub-problema de programação

linear inteira equivalente:

- com o algoritmo construtivo `CEV` (`3fases_II + CEV`)
- com o algoritmo construtivo `CEO_II` (`3fases_II + CEO_II`)
- com o algoritmo construtivo `CEV` e a busca local substituída pela resolução exata do modelo de programação inteira (`3fases_II + CEV + MIP`)
- com o algoritmo construtivo `CEO_II` e a busca local substituída pela resolução exata do modelo de programação inteira (`3fases_II + CEO_II + MIP`)

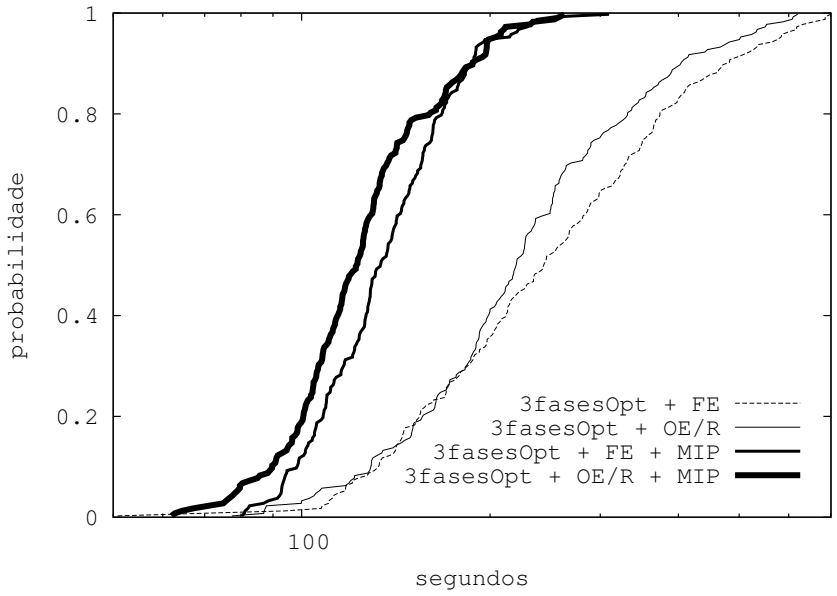

Figura 3.4: 65 localidades e padrão P_0 (alvo: 364).

Estas quatro variantes foram selecionadas de forma a permitir um estudo do impacto causado pelo algoritmo guloso com ênfase no objetivo e pela busca local por programação inteira nos resultados da heurística de três fases, tanto separadamente como combinados.

Os gráficos mostram que tanto o algoritmo construtivo `CEO_II` quanto a resolução exata tiveram impacto positivo e resultaram em versões mais eficientes do que a versão original. A versão completa com ambas extensões (`3fases_II + CEO_II + MIP`) mostrou que encontra o alvo com maior probabilidade para um dado limite de tempo. Além disso, esta variante apresentou também uma variabilidade menor nos tempos de processamento para as instâncias e alvos selecionados para este experimento, comparando-se com a heurística original `3fases_II + CEV`.

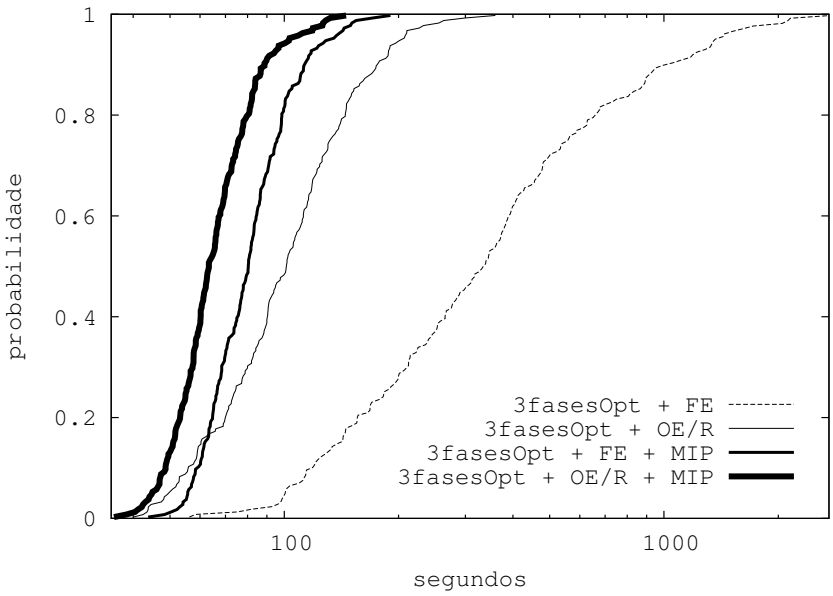

Figura 3.5: 65 localidades e padrão P_1 (alvo: 837).

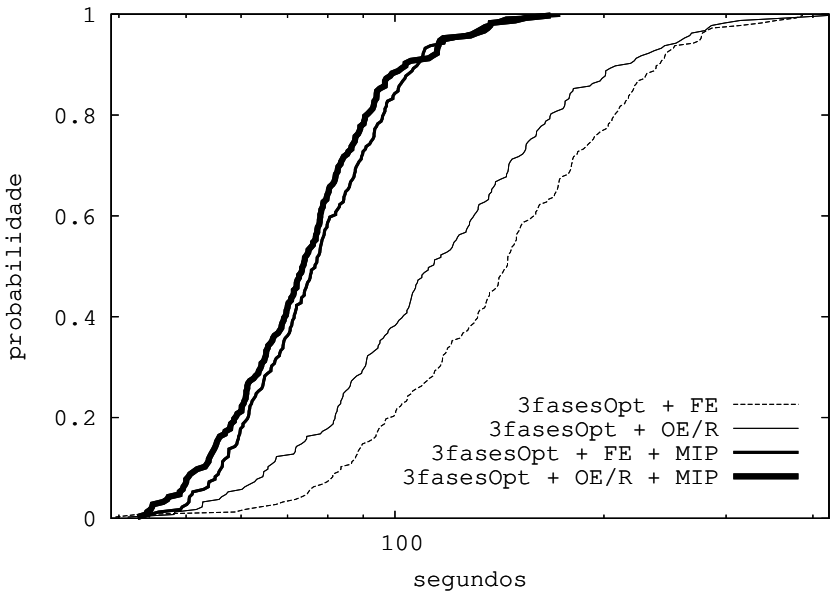

Figura 3.6: 85 localidades e padrão P_0 (alvo: 514).

3.4.4 Contribuição das componentes da heurística

As Tabelas 3.8 a 3.10 apresentam os resultados de um experimento que ilustra a contribuição de cada componente da variante `3fases_II + CEO_II + MIP`, que teve melhor desempenho nos experimentos anteriores. Para cada instância são exibidos os valores médios do tempo de processamento e da função objetivo, sobre dez execuções, para as melhores soluções encontradas em quatro momentos da execução da heurística: (1) a primeira solução viável encontrada, (2) dentro da heurística aprimorante, após a resolução exata

inicial aplicada a todas as localidades (separadamente), (3) após a primeira perturbação seguida da resolução exata somente dos sub-problemas associados às localidades envolvidas, ainda dentro da heurística aprimorante baseada em ILS e (4) após 1000 segundos de execução. A última linha de cada tabela traz os valores médios normalizados, em relação à solução viável inicial, para o valor da função objetivo após cada fase, ilustrando a sucessiva melhora na solução resultante após a aplicação de cada um dos componentes da heurística.

Instância	Solução inicial		MIP inicial		ILS 1^a iteração		ILS 1000 s	
id.	tempo (s)	valor	tempo (s)	valor	tempo (s)	valor	tempo (s)	valor
I_1	0,90	702,6	1,59	611,6	0,17	607,0	662,86	343,8
I_2	1,35	787,4	1,46	695,8	0,22	687,6	727,34	418,8
I_3	0,55	732,0	1,48	652,4	0,15	646,8	787,26	334,4
I_4	0,78	745,6	1,43	646,4	0,20	640,6	787,71	334,4
I_5	1,03	747,0	1,44	655,4	0,14	651,0	723,42	356,0
Média	0,92	742,9	1,48	652,3	0,18	646,6	737,72	357,5
Normalizado	-	1,00	-	0,88	-	0,87	-	0,48

Tabela 3.8: Contribuição dos componentes da heurística `3fases_II+CEO_II + MIP` (65 localidades e padrão P_0).

Instância	Solução inicial		MIP inicial		ILS 1^a iteração		ILS 1000 s	
id.	tempo (s)	valor	tempo (s)	valor	tempo (s)	valor	tempo (s)	valor
I_1	2,60	1085,8	1,41	1013,8	0,20	1008,2	764,31	806,0
I_2	2,20	1128,4	1,46	1048,0	0,26	1042,4	751,51	854,6
I_3	0,08	1000,0	1,39	970,8	0,16	966,0	593,96	777,4
I_4	1,22	998,6	1,38	934,8	0,15	929,6	714,22	688,4
I_5	1,37	1053,2	1,38	982,4	0,24	976,8	581,88	758,4
Média	1,49	1053,2	1,40	990,0	0,20	984,6	681,18	777,0
Normalizado	-	1,00	-	0,94	-	0,93	-	0,74

Tabela 3.9: Contribuição dos componentes da heurística `3fases_II+CEO_II + MIP` (65 localidades e padrão P_1).

3.4.5 Abordagem exata vs. aproximada

O último experimento deste capítulo foi realizado de forma a permitir uma comparação dos resultados obtidos pela heurística de três fases `3fases_II + CEO_II + MIP` (a variante com os melhores resultados) com aqueles encontrados pela resolução exata apresentada no Capítulo 2 para algumas instâncias

Instância	Solução inicial		MIP inicial		ILS 1^a iteração		ILS 1000 s	
id.	tempo (s)	valor	tempo (s)	valor	tempo (s)	valor	tempo (s)	valor
I_1	23,80	945,0	1,51	831,2	0,10	825,8	797,47	487,8
I_2	31,80	957,4	1,56	838,2	0,08	833,4	788,43	502,8
I_3	11,62	967,0	1,52	852,8	0,16	848,8	862,93	482,8
I_4	26,99	947,0	1,59	819,0	0,09	815,2	768,78	435,6
I_5	25,79	924,4	1,49	812,8	0,16	806,6	825,71	399,4
Média	24,00	948,2	1,53	830,8	0,12	826,0	808,66	461,7
Normalizado	-	1,00	-	0,88	-	0,87	-	0,49

Tabela 3.10: Contribuição dos componentes da heurística `3fases_II+CEO_II + MIP` (85 localidades e padrão P_0).

teste. Selecionaram-se apenas os resultados relativos à resolução com o Modelo 3, que foi o que proporcionou os menores tempos para a resolução exata das instâncias consideradas através do CPLEX.

Os resultados numéricos são apresentados nas Tabelas 3.11 a 3.13. São apresentados os resultados relativos às maiores instâncias que foram resolvidas pela abordagem exata utilizando o Modelo 3, com até 120 partidas e 240 árbitros. Exibe-se inicialmente o número de localidades da instância (ℓ), o padrão usado em sua geração e sua identificação. As duas colunas subseqüentes mostram o tempo gasto (em segundos) e o valor da solução ótima encontrada pelo CPLEX. As células assinaladas com "memória" indicam que o CPLEX não conseguiu resolver o problema por insuficiência de memória. O símbolo '$\circ$' aparece ao lado dos valores das soluções, nos casos em que o resolvedor não provou a otimalidade da melhor solução encontrada dentro do limite de 7200 segundos. O símbolo '$\bullet$' é exibido nos casos em que o resolvedor não encontrou solução viável alguma. Neste caso, a célula em questão representa o valor da melhor solução encontrada pela heurística.

As demais colunas trazem os resultados obtidos pela variante `3fases_II + CEO_II + MIP`. Os resultados apresentados consistem primeiramente nas diferenças percentuais entre os valores das soluções obtidas pela heurística, em diferentes momentos da execução, e o valor da melhor solução encontrada (exibido na quinta coluna). As primeiras quatro colunas com resultados da heurística apresentam os valores médios (sobre as dez execuções) das soluções encontradas após dez segundos ($\overline{\Delta}_{10}$), 30 segundos ($\overline{\Delta}_{30}$), 60 segundos ($\overline{\Delta}_{60}$) e ao final do limite de tempo ($\overline{\Delta}_f$). A penúltima coluna (Δ^*) exibe o valor da diferença percentual entre o valor da melhor solução encontrada pela heurística (dentre as dez execuções) e o valor exibido na quinta coluna. A última coluna exibe o tempo médio gasto pela heurística para encontrar a melhor solução de

Instância			Ótimo		Heurística					
ℓ	padrão	id.	tempo (s)	valor	$\overline{\Delta}_{10}$	$\overline{\Delta}_{30}$	$\overline{\Delta}_{60}$	$\overline{\Delta}_{f}$	Δ^*	tempo (s)
8	P_0	I_1	129,69	70	17,4	12,0	8,6	2,6	0,0	71,60
		I_2	79,39	101	2,8	0,0	0,0	0,0	0,0	17,63
		I_3	176,53	78	0,0	0,0	0,0	0,0	0,0	7,04
		I_4	227,59	99	2,8	0,0	0,0	0,0	0,0	8,93
		I_5	77,40	87	6,7	2,3	0,7	0,2	0,0	45,83
8	P_1	I_1	98,34	155	0,8	0,0	0,0	0,0	0,0	11,61
		I_2	69,09	115	4,2	1,7	0,3	0,2	0,0	43,71
		I_3	59,78	138	5,8	3,9	2,5	1,2	0,0	82,72
		I_4	38,50	186	1,7	0,4	0,4	0,0	0,0	46,78
		I_5	61,04	169	2,0	0,7	0,4	0,0	0,0	45,92
12	P_0	I_1	14,31	136	6,2	3,2	2,2	1,0	0,0	86,27
		I_2	18,54	117	9,9	5,3	3,8	3,1	1,7	65,23
		I_3	21,20	78	11,8	5,9	3,3	2,1	0,0	59,69
		I_4	7200,00	∘ 105	9,7	6,1	3,8	2,1	1,9	97,66
		I_5	8,83	107	6,4	3,6	2,8	1,3	0,0	59,81

∘ Solução ótima não foi provada.

Tabela 3.11: Instâncias com 80 partidas e 160 árbitros, com o tempo para a heurística limitado em 180s.

Instância			Ótimo		Heurística					
ℓ	padrão	id.	tempo (s)	valor	$\overline{\Delta}_{10}$	$\overline{\Delta}_{30}$	$\overline{\Delta}_{60}$	$\overline{\Delta}_{f}$	Δ^*	tempo (s)
10	P_0	I_1	85,65	124	7,6	2,7	1,5	0,0	0,0	110,34
		I_2	92,89	125	3,2	0,0	0,0	0,0	0,0	17,17
		I_3	272,46	86	8,1	2,1	2,1	0,0	0,0	24,38
		I_4	320,65	135	3,6	0,1	0,0	0,0	0,0	18,85
		I_5	153,65	84	18,6	11,0	8,1	4,3	2,4	120,03
10	P_1	I_1	73,56	209	3,8	1,9	1,2	0,6	0,0	115,47
		I_2	279,04	158	6,6	2,9	1,0	0,3	0,0	93,20
		I_3	388,98	153	9,8	5,0	3,4	1,4	1,3	158,28
		I_4	106,05	182	6,3	2,2	1,0	0,0	0,0	125,37
		I_5	61,22	251	6,6	3,9	3,2	1,5	0,8	157,72
15	P_0	I_1	30,71	156	6,9	3,1	1,9	0,6	0,0	111,95
		I_2	72,28	125	14,7	7,4	5,3	1,3	0,0	155,92
		I_3	7190,56	108	13,1	6,7	4,8	3,0	1,9	108,21
		I_4	37,14	149	8,6	5,0	3,8	1,5	0,0	117,51
		I_5	7200,00	∘ 130	8,9	4,2	2,5	0,9	0,0	135,87

∘ Solução ótima não foi provada.

Tabela 3.12: Instâncias com 100 partidas e 200 árbitros, com o tempo para a heurística limitado em 240s.

cada uma das dez execuções.

Observa-se que a heurística foi capaz de encontrar, ao longo das dez execuções, uma solução ótima ($\Delta^* = 0,0$) para 34 de um total de 40 instâncias

para as quais o ótimo foi provado pelo CPLEX. Além disso, em apenas dez segundos a heurística obteve soluções com diferenças percentuais médias ($\overline{\Delta}_f$) menores que 10% em 31 casos. Para 29 instâncias, a diferença percentual média ficou abaixo dos 5% após 30 segundos de execução, considerando-se novamente as instâncias para as quais a solução ótima foi encontrada com algum dos modelos. A diferença percentual média ($\overline{\Delta}_f$) ficou abaixo de 5% para todas as instâncias.

Estes resultados ilustram a eficiência da abordagem aproximada proposta, que é capaz de encontrar boas soluções mesmo com limites reduzidos de tempo. Finalmente, observa-se que a abordagem aproximada sempre encontrou alguma solução viável, ao contrário da abordagem exata que, dentro do limite de 7200 segundos, não foi capaz de encontrar solução inteira viável alguma para as instâncias I_2 e I_3 com 120 partidas, 12 localidades e padrão P_1 (Tabela 3.13).

Instância			Ótimo		Heurística					
ℓ	padrão	id.	tempo (s)	valor	$\overline{\Delta}_{10}$	$\overline{\Delta}_{30}$	$\overline{\Delta}_{60}$	$\overline{\Delta}_f$	Δ^*	tempo (s)
12	P_0	I_1	285,11	152	11,8	8,7	5,7	1,4	0,0	160,16
		I_2	468,85	131	1,8	0,0	0,0	0,0	0,0	14,23
		I_3	287,48	129	0,2	0,0	0,0	0,0	0,0	8,37
		I_4	797,89	159	7,2	0,5	0,0	0,0	0,0	30,64
		I_5	655,76	124	9,0	3,9	2,3	1,5	0,0	73,16
12	P_1	I_1	151,1	276	6,1	2,8	1,8	0,7	0,7	188,74
		I_2	memória	• 181	11,0	6,0	3,5	0,7	0,0	223,64
		I_3	memória	• 209	8,6	4,4	2,2	0,4	0,0	143,23
		I_4	197,26	284	5,7	2,9	1,6	0,4	0,0	145,75
		I_5	177,32	212	6,6	3,2	1,9	0,3	0,0	132,90
18	P_0	I_1	7200	∘ 160	13,0	7,5	4,4	1,3	0,0	178,92
		I_2	272,1	131	11,6	3,7	0,9	0,0	0,0	64,36
		I_3	441,88	129	16,4	6,7	3,9	0,0	0,0	199,69
		I_4	132,77	159	13,0	6,3	3,6	0,0	0,0	231,57
		I_5	134,91	170	6,1	2,9	1,6	0,1	0,0	165,17

∘ Solução ótima não foi provada.

• Valor da melhor solução encontrada pela heurística.

Tabela 3.13: Instâncias com 120 partidas e 240 árbitros, com o tempo para a heurística limitado em 360s.

3.5 Conclusões

Neste capítulo foi proposta uma abordagem de resolução aproximada em três fases para o problema de atribuição de árbitros. O objetivo destes algoritmos é permitir a resolução de instâncias de tamanho real que não podem

ser resolvidas de maneira exata pelo resolvedor de programação linear inteira utilizado.

Três diferentes algoritmos construtivos foram propostos e seus resultados comparados. Experimentos mostraram a eficiência de tais heurísticas em obter boas soluções iniciais com baixos tempos de processamento.

O grande número de restrições, que impedem muitas das atribuições de árbitros a PA's, torna não trivial o problema de encontrar uma atribuição viável. Os resultados computacionais das heurísticas construtivas mostraram que mesmo a variante que prioriza apenas a viabilidade não é capaz de encontrar sempre uma solução viável. Por este motivo, foi proposta uma heurística reparadora baseada na metaheurística ILS que se mostrou capaz de viabilizar todas as soluções iniciais inviáveis obtidas pelos algoritmos construtivos. Esta heurística conseguiu também, em diversos casos, obter soluções viáveis mesmo partindo de soluções iniciais geradas de forma completamente aleatória. Além disso, observa-se pelos resultados que as variantes das heurísticas construtivas que priorizam o objetivo obtiveram melhores resultados que aquela que visa somente a viabilidade. Estes resultados foram atingidos também devido à eficiência da heurística reparadora em tornar viáveis as soluções construídas pelas heurísticas que priorizam o objetivo.

Foi ainda proposta uma heurística aprimorante, também baseada na metaheurística ILS, com o objetivo de melhorar iterativamente as soluções viáveis obtidas pelas heurísticas construtivas e reparadora. Uma abordagem híbrida que substitui o procedimento de busca local pela resolução exata de um modelo de programação inteira, foi também utilizada e comparada com a abordagem de busca local por vizinhanças. Os resultados de diversos experimentos computacionais foram discutidos e mostraram a eficiência das abordagens propostas. Em particular, a variante que substituiu o procedimento de busca local pela resolução exata obteve os melhores resultados. A extensão desta abordagem para outras metaheurísticas é simples e os resultados promissores mostram que a hibridização de metaheurísticas com algoritmos exatos pode levar a algoritmos mais rápidos e robustos e deve ser mais explorada.

Concluindo, as heurísticas propostas se mostraram bastante eficientes e capazes de encontrar soluções ótimas para as instâncias de pequeno e médio portes em tempos computacionais razoáveis. Foram também capazes de resolver instâncias de tamanho real que não puderam ser tratadas pela abordagem exata com o resolvedor utilizado. No próximo capítulo são discutidas algumas variantes ao problema básico formulado no Capítulo 2 e propostos algoritmos para uma variante do PAA com outra função objetivo.