Capítulo 12

Algoritmo Evolutivo Híbrido para Escalonamento Integrado na Agroindústria

Ademir Aparecido Constantino*, Dario Landa-Silva e Wesley Romão

Resumo: Este capítulo aborda um problema envolvendo o escalonamento integrado de pessoal e caminhões em uma agroindústria brasileira que processa 1 milhão de aves por semana. O escalonamento deve assegurar o constante fornecimento de aves para a fábrica a fim de evitar a ociosidade. O objetivo é minimizar a quantidade de horas extras pagas para os catadores e o tempo de espera dos caminhões no pátio da fábrica. Propõe-se um algoritmo evolutivo combinado com busca local e simulação discreta determinística. Os experimentos foram executados com dados reais e os resultados mostram que o método é efetivo para produzir escalas robustas.

Palavras-chave: Escalonamento, Agroindústria, Algoritmos genéticos.

Abstract: *This chapter addresses a problem involving integrated personnel and lorry scheduling in a Brazilian farming industry that processes around 1 million poultries per week. The schedule should ensure constant supply of poultries to the factory to avoid idle processing time. The objective is minimizing the extra time paid to squads and minimizing the waiting time of lorries at the factory parking. We propose an evolutionary algorithm combined with local search and deterministic discrete simulation. The experiments were done with real data and the results show that the method is effective to produce robust schedules.*

Keywords: *Scheduling, Farming industry, Genetic algorithm.*

*Autor para contato: aaconstantino@uem.br

1. Introdução

Escalonamento *(scheduling)* de atividades significa designar um conjunto de eventos ao longo do tempo, atendendo a um conjunto de restrições e, em geral, otimizando alguma função objetivo. Os problemas de escalonamento desempenham um papel importante dentro da computação, especialmente dentro da pesquisa operacional, pelo fato de fomentar a investigação de modelos computacionais e de novas técnicas algorítmicas.

O problema de escalonamento de atividades pode ser encontrado em diversos contextos, seja envolvendo máquinas (simples ou paralela) (Sheen & Liao, 2007; Blazewicz et al., 1994), processos (Sonmez & Gursoy, 2007; Bansal et al., 2009) ou pessoas (Yunes et al., 2000). O escalonamento de pessoal pode ser dividido em dois tipos: escalonamento de pessoal em local fixo e escalonamento de pessoal no setor de transportes (Bodin, 1983). O escalonamento no setor de transportes é um dos mais complexos do ponto de vista computacional (Bodin, 1983). O desafio computacional, portanto, é encontrar uma metodologia que obtenha uma solução satisfatória para o problema em tempo computacional viável.

Como reportado por Bodin (1983), a maioria dos problemas de escalonamento não admitem solução em tempo polinomial e são poucos os casos modelados por programação matemática utilizando algoritmos exatos (Yunes et al., 2000). O escalonamento na agroindústria tratado aqui se diferencia do escalonamento de transporte pelo fato da hora início de cada viagem ser desconhecido. O problema investigado surge no contexto de uma indústria que processa mais de 1 milhão de aves por semana onde pequenas melhorias no escalonamento podem resultar em redução significativa do desperdício, implicando em redução de custos. O escalonamento automatizado deve ser capaz de definir a escala de atividades das equipes de apanha, assegurando o constante fornecimento de suprimento para a indústria, respeitando um conjunto de restrições operacionais e otimizando uma função objetivo. Além disto, informações sobre o tamanho da fila de caminhões ou tempo de espera na fila são parâmetros que devem ser otimizados. Por outro lado, estas informações são complexas para serem integradas em um modelo de programação matemática. Considerando a complexidade de se criar um modelo de programação matemática e a dificuldade de incorporar informações das filas, optou-se por utilizar um algoritmo heurístico.

O problema em tela não tem sido reportado na literatura, exceto uma variação do mesmo apresentado por Hart et al. (1999), também baseado em um caso real no norte da Inglaterra. Neste caso, a empresa não desejava um escalonamento otimizado, mas apenas um sistema que automatizasse o escalonamento, minimizando o número de violações de restrições. Portanto, o caso foi tratado como um problema de satisfação de restrições, o qual foi abordado usando algoritmos genéticos. Porém várias caracterís-

ticas se diferenciam do problema investigado neste estudo. Na Seção 2 o problema será apresentado em detalhes, fazendo uma comparação com o trabalho de Hart et al. (1999).

Dada a complexidade para modelar o problema investigado e a dificuldade de calcular as informações da fila de espera, optou-se por implementar uma metodologia baseada em algoritmos genéticos, integrando-o à programação matemática e à técnica de simulação discreta determinística para cálculo das estatísticas das filas. O algoritmo possui duas fases hierárquicas, sendo que a primeira fase procura minimizar o custo com as equipes de apanha, enquanto que a segunda fase tenta melhorar o resultado focando na minimização do tamanho da fila de espera.

Algoritmos genéticos também tem sido utilizados em diversos problemas de escalonamento, seja de pessoal (Wren & Wren, 1995; Kotecha et al., 2004), de máquinas (Zhang et al., 2008; Park, 2001) e de processos (Contreras et al., 2005).

2. Descrição do Problema

O objeto deste estudo é o escalonamento integrado na agroindústria. Este problema está relacionado ao Escalonamento de Pessoal *(Personnel Scheduling)* integrado com escalonamento da produção e designação de veículos de transporte. O problema investigado surge de um caso real observado no setor da agroindústria, cujo insumo processado é oriundo de carga de aves vivas que não podem ser estocadas por muito tempo. A empresa investigada realiza o processamento de mais de um milhão de aves por semana, dividido em três plantas (fábricas) em diferentes localizações. Os caminhões com carga que chegam na fábrica devem aguardar num galpão ventilado, principalmente nos períodos de alta temperatura do verão. O desafio diário da empresa é realizar o escalonamento das equipes de apanha (mais de 100 pessoas no total) para manter as fábricas em constante funcionamento, atendendo às restrições operacionais e trabalhistas, além de minimizar os custos. A Figura 1 resume um exemplo de escalonamento com duas equipes de apanha e mostra a sequência de locais para as equipes e as designações (transportes) das cargas para três fábricas. A ilustração mostra duas sequências individuais, uma para cada equipe, embora em alguns momentos o mesmo local possa ser compartilhado por duas ou mais equipes, não necessariamente no mesmo tempo, pois cada equipe terá sua escala com horários possivelmente diferentes. Nota-se, também, que as escalas dos caminhões com as cargas têm sua origem e destino nas fábricas. As designações dos caminhões são para o transporte de cargas de cada granja para a fábrica. Uma granja pode ter várias cargas que podem ter destinos (fábricas) diferentes.

As aves devem ser coletadas em diferentes granjas, em locais espalhados geograficamente num raio de aproximadamente 150 Km de cada fábrica. A

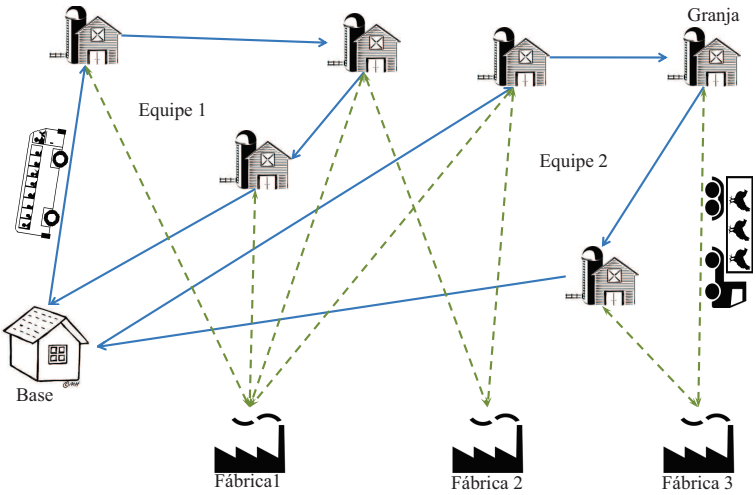

Figura 1. Ilustração do escalonamento de duas equipes de apanha das cargas para três fábricas. As linhas contínuas representam a sequência de locais visitados por cada equipe. As linhas tracejadas representam as designações de cargas das granjas para as fábricas.

cada dia novos locais de coleta das aves são definidos, implicando na necessidade de realizar escalonamentos diários. O pessoal que realiza a apanha das aves é dividido em várias equipes que podem ser designadas para diferentes locais durante o dia. Logo, o tempo de transporte entre os locais de apanha deve ser minimizado. As equipes têm sua base em um local diferente das fábricas. Algumas informações, tais como a sequência de locais de apanha, a hora de início e término das atividades, a hora do carregamento e descarregamento dos caminhões e o destino de cada carregamento (carga) devem ser descobertas com a definição do escalonamento.

Para obter este escalonamento, algumas regras devem ser levadas em consideração, como o acordo com a empresa, as regras trabalhistas e restrições operacionais. Além disto, há uma função objetivo a ser otimizada relacionada com custos operacionais.

Os custos operacionais incluem custos com horas extras para as equipes de apanha, custos com horas ociosas das equipes, custos com transporte e custos relacionados ao tempo de espera do caminhão no pátio da empresa. Considerando o alto custo da ociosidade da indústria, este custo é considerado como uma restrição do problema. Assim, a função objetivo se resume à minimização de custos com as equipes de apanha (horas pagas) e ao tempo de espera na fila (pátio da fábrica).

Para efeito de cálculo do tempo de trabalho pago às equipes, são considerados o tempo de viagem e o tempo gasto durante a apanha e carregamento dos caminhões. Além disto, algumas restrições (regras) devem ser atendidas na elaboração das escalas de trabalho. As regras a serem seguidas são as seguintes:

- O tempo normal de trabalho são 7 horas e 20 minutos (excluindo o tempo de descanso). O tempo que extrapolar é considerado hora extra, incidindo um acréscimo de 50% sobre o valor pago no tempo normal.
- Se o total de horas trabalhadas for superior a 4 horas e não exceder 6 horas, deverá ser adicionado um tempo de descanso de, no mínimo, 15 minutos.
- Se o total de horas trabalhas for superior a 6 horas, deverá ser adicionado um descanso de no mínimo 1 hora.
- O descanso deve ser de, no máximo, 2 horas, pois acima deste tempo será considerado tempo pago. A hora extra também é contabilizada sobre o tempo mínimo de descanso caso não seja considerado na escala.

Além das restrições operacionais, existem alguns dados gerais que devem ser utilizados pelo algoritmo, tais como: localização de todas as granjas, localização da base das equipes de apanha, localização das fábricas, distância entre todos os locais, velocidade média dos caminhões, velocidade de abate de cada fábrica, hora de início de operação de cada fábrica, hora de término de operação de cada fábrica e capacidade de cada caminhão.

O escalonamento é realizado diariamente e os dados de entrada do algoritmo, específicos para cada dia, devem ser: as granjas onde serão coletadas as aves, e o número de aves de cada granja.

Com base nas informações apresentadas sobre o problema em estudo, a Tabela 1 resume os aspectos em comum e as diferenças do presente trabalho com o de Hart et al. (1999). Observa-se na tabela que o problema em estudo possui mais diferenças do que aspectos em comum. Hart et al. (1999) tem como objetivo minimizar as violações de restrições, enquanto que nesta investigação o objetivo é miminizar custos e tempo na fila, satisfazendo as restrições. O fato de haver uma flexibilidade para construção dos turnos de trabalho, com divisões de tarefas de uma hora de duração, significa que há mais possibilidades de combinações de tarefas, ampliando, assim, a complexidade computacional para resolver o problema. A limitação de espaço no pátio de espera da fábrica e o tempo de espera no pátio são duas características que estão relacionadas e que impõem tanto uma restrição ao problema como um objetivo a ser otimizado. Calcular o tempo de espera

no pátio (fila) é outro aspecto que também introduz um novo desafio para a resolução do problema.

Tabela 1. Comparação com o trabalho de Hart et al. (1999).

	Problema investigado	**Hart, Ross & Nelson (1999)**
Objetivo	Minimizar os custos operacionais e tempo na fila	Minimizar o número de violações de restrições
Abordagem	Problema de Otimização	Problema de Satisfação de Restrição
Número de fábricas	Três fábricas em locais diferentes e com velocidades diferentes de abate	O número e localização não são informados.
Base das equipes	Uma localização diferente das fábricas	No mesmo local da fábrica
Turno de trabalho	Flexível, com divisões de tarefas (tempo de carregamento) em intervalos de uma hora	Os turnos são divididos em três tipos: manhã, tarde, flutuante
Transporte das aves	Terceirizado. Neste caso as escalas dos motoristas não são de responsabilidade da indústria	Frota própria, mas utiliza serviços de terceiros quando necessário
Pátio de espera na fábrica	Cada fábrica tem um número fixo de vagas em um galpão ventilado	Não consta essa informação
Tempo de espera no pátio da fábrica	Não há um limite de tempo máximo de espera *	Restrições governamentais estabelecem um limite de tempo

* Não há limite de tempo máximo, porém é desejável que a carga permaneça em espera o mínimo de tempo possível para reduzir o custo operacional de manter a carga no pátio de espera e reduzir a perda de aves.

3. Modelagem Matemática

Como resultado da divisão do número de aves de uma granja pela capacidade de transporte de cada veículo, tem-se o que é denominado de "carga". Cada carga está associada as seguintes informações:

1. Local de origem (granja);
2. Local de destino (fábrica);
3. Hora de início do carregamento;
4. Duração do carregamento;
5. Equipe de apanha responsável;
6. Número de aves.

Uma vez conhecido o conjunto de granjas e suas informações, como localização e número de aves, então o número de cargas, as suas origens e a duração para carregamento são obtidos automaticamente. Portanto, resta descobrir o destino (fábrica), a equipe responsável pelo carregamento e a hora de início do carregamento. Todos estes dados são associados a cada carga. Uma etapa fundamental para minimização dos custos é descobrir a associação das cargas às fábricas. Para isto foi criado um modelo de programação linear inteira binária descrito a seguir:

$$Min \sum_{g=1}^{ng} \sum_{f=1}^{nf} d_{gf} x_{gf} \tag{1}$$

$$Sujeito\ a: \quad \sum_{f=1}^{nf} x_{gf} = 1;\ g = 1, ..., ng; \tag{2}$$

$$\sum_{g=1}^{ng} s_g x_{gf} \leq cp_f;\ f = 1, ..., nf \tag{3}$$

$$x_{gf} \in \{0, 1\},\ g = 1, ..., ng;\ f = 1, ..., nf$$

onde ng é o número de cargas, nf é o número de fábricas, d_{gf} é a distância da granja que originou a carga g ($g = 1, ..., ng$) até a fábrica f ($f = 1, ..., nf$), s_g é o número de aves da carga g e cp_f é a capacidade de processamento (em número de aves) da fábrica f. Por fim, x_{gf} é a variável de decisão, sendo que $x_{gf} = 1$ se a carga g for alocada à fábrica f, sendo nf o número de fábricas. A restrição 2 garante que apenas uma carga seja alocada a uma única fábrica, enquanto que a restrição 3 garante que a capacidade da fábrica seja respeitada. A função objetivo dada pela Equação 1 procura minimizar a soma total das distâncias relacionadas com o transporte das aves.

Este modelo de programação matemática resolve apenas uma parte do problema global, que diz respeito ao destino das cargas de forma a minimizar custos de transporte. A geração de escala de trabalho por modelos de programação matemática ainda é um desafio para os pesquisadores da área (Bodin, 1983), o que tem justificado o uso de algoritmos heurísticos.

Neste problema em estudo foram identificados dois aspectos que dificultam a formulação de um modelo geral de programação matemática para o problema global:

1. Conceder período de descanso conforme as regras trabalhistas previamente definidas. Neste caso em particular, sempre que possível o descanso deve ser durante o tempo de transporte entre granjas ou da granja para sua base. Esta característica surge neste contexto pelo fato do tempo de transporte ser contabilizado como tempo de

trabalho, fato que não ocorre em escalonamento de pessoal em local fixo como, por exemplo, escalonamento de tele-atendentes.

2. A função objetivo deve considerar o tamanho médio das filas de caminhões nos pátios das fábricas. O uso de simulação tem sido uma alternativa para superar esta dificuldade de introduzir cálculos estatísticos de filas em modelos de programação matemática. Esta abordagem pode ser observada em Lin et al. (2000), Harrison & Zeevi (2005) e Atlason et al. (2008), os quais utilizam a combinação de programação linear e simulação na resolução de problemas de escalonamento no setor de tele-atendimentos.

Por estas dificuldades de modelagem matemática do problema, a implementação de um algoritmo heurístico passa a ser uma alternativa interessante. Neste trabalho é proposto um algoritmo heurístico que utiliza o modelo de designação linear em duas fases: fase de construção da solução e fase de melhoramento (busca local). A formulação do problema de designação (PD), denotada por $PD([c_{ij}])$, é a seguinte:

$$z = Min \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} c_{ij} x_{ij} \tag{4}$$

$$Sujeito\ a: \quad \sum_{i=1}^{n_1} x_{ij} = 1,\ \forall j = 1, ..., n_2 \tag{5}$$

$$\sum_{j=1}^{n_2} x_{ij} = 1,\ \forall i = 1, ...n_1 \tag{6}$$

$$x_{ij} \in \{0, 1\},\ \forall i = 1, ..., n_1,\ \forall j = 1, ..., n_2$$

A variável de decisão x_{ij} é binária $\{0, 1\}$, sendo que $x_{ij} = 1$ se o item i for associado com o item j. Caso contrário, $x_{ij} = 0$, e c_{ij} é o custo de associar o item i ao item j. Os itens podem ter diferentes significados de acordo com a fase de resolução (construção e melhoramento), apresentadas nas seções seguintes. O valor de n_1 deve ser igual a n_2. Porém, na fase de construção da solução n_1 representa o número de equipes (reais e fictícias) e n_2 o número de cargas ng. Considerando que o número de cargas é sempre superior ao número de equipes, são criadas equipes fictícias para tornar a matriz $[c_{ij}]$ quadrada. Na fase de melhoramento, n_1 e n_2 serão sempre o número máximo de equipes (ne_{max}) permitidas para o escalonamento. Embora o algoritmo utilize o número máximo de equipes disponíveis, deve-se observar que a abordagem procura automaticamente minimizar o número de equipes, tendo em vista que o objetivo é minimizar o custo global com pessoal.

4. O Algoritmo Proposto

O método proposto é um algoritmo evolutivo que utiliza alguns procedimentos construtivos e de busca local (melhoramento). Nesta seção serão apresentadas a forma de representação da solução, os procedimentos construtivos, o procedimento de busca local e o procedimento geral baseado em algoritmos genéticos.

Uma escala de trabalho para uma equipe é uma sequência de atividades a ser executada ao longo do dia. Porém, o tempo de carregamento de um caminhão é de aproximadamente uma hora. Portanto, a escala de trabalho pode ser discretizada em uma sequência de atividades com duração de uma hora. Cada atividade está relacionada com uma carga definida anteriormente. Para efeito de simplificação do problema, a duração do carregamento de todas as cargas será considerado o intervalo de uma hora. Assim, considere t o índice de tempo ao longo do dia de trabalho, sendo $t = 1, ..., 24$, onde $t = 1$ significa o primeiro horário do dia em que alguma equipe inicia o trabalho de carregamento.

4.1 Procedimento construtivo

Dado um número máximo de equipes ne_{max} pré-definido, a fase de construção da solução inicial parte do princípio de que a sequência de atividades (cargas) de cada equipe ainda está vazia e, então, as cargas são designadas sucessivamente até que todas as cargas sejam atribuidas para o número máximo de equipes. A idéia do algoritmo é atribuir as cargas para as equipes considerando que todos os carregamentos iniciam em $t = 1$. Assim, em $t = 1$ todas as equipes recebem a designação de uma carga selecionada aleatoriamente. Para os demais intervalos de tempo t $(t > 1)$ a designação é resolvida utilizando um problema de designação, o qual está associado a uma matriz de custo c_{ij}, de dimensão ng, relacionada ao custo de associar uma carga j a uma equipe i. Como o número de cargas é sempre maior do que o número de equipes (ne_{max}), então a matriz é dividida em dois blocos e assume valores conforme definido na Figura 2.

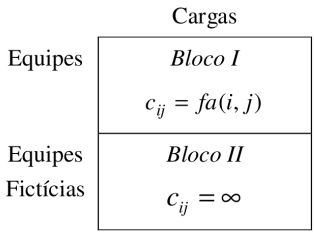

Figura 2. Estrutura da matriz de custo para a fase de construção da solução.

Onde os custos do bloco I da matriz são calculados pela seguinte função:

$$fa(i,j) = hp + ph, \tag{7}$$

sendo hp as horas pagas para a equipe i ao ser designada ao trabalho de carregamento da carga j, e ph uma penalidade caso alguma restrição não seja atendida. Para o cálculo das horas pagas são levadas em considerações as regras apresentadas na Seção 2. Como penalidade foi considerado o limite de 9 horas de trabalho contínuo. O algoritmo do procedimento de construção é apresentado na Figura 3.

Procedimento *Construção*
Início
$t := 1$;
Para e=1 **até** ne_{max} **faça**:
Selecionar aleatoriamente uma granja e atribuir uma carga para a equipe e;
Associar a hora de início do carregamento igual a t;
Enquanto houver cargas não atribuídas **faça:**
$t := t+1$;
Construir a matriz de custo para o horário t, $[c_{ij}]$;
Resolver o problema $PD([c_{ij}])$;
Associar as cargas para as equipes conforme a solução de $PD([c_{ij}])$;
Associar a hora de início do carregamento igual a t;
Fim.

Figura 3. Procedimento de construção.

4.2 Busca local baseada em custo

A busca local tem o objetivo de realizar pesquisa em torno de uma solução corrente (algumas vezes chamada de busca na vizinhança) na tentativa de encontrar soluções aprimoradas. Neste trabalho propõe-se dois procedimentos de busca que são baseados em VND (*Variable Neighborhood Descent*), que é uma particularidade do VNS (*Variable Neighborhood Search*) (Hansen & Mladenović, 2001).

Um procedimento de busca local, denominado k-*swap*, consiste em selecionar k intervalos de tempo consecutivos iniciando no tempo t; para cada equipe são agrupadas as atividades associadas neste intervalo gerando os "blocos de atividades" denotados por $ba(k,t)$. O k-*swap* realiza a investigação de troca de blocos de atividades destes intervalos entre as escalas das equipes.

A investigação de busca do *k-swap* se dá pela resolução de sucessivos problemas de designação. Cada investigação requer uma matriz de custo $[c_{ij}]$, de dimensão ne_{max}, que representa o custo de associar um bloco de atividades i com a escala da equipe j. A Figura 4 ilustra a estrutura da matriz de custo para cada problema de designação utilizado pelo *k-swap*. Nota-se que, diferente da estrutura da matriz do procedimento de construção, agora a matriz de custo trata da relação de custos em associar escalas ou parte das escalas, enquanto que a matriz anterior relacionava o custo de associar cargas às equipes.

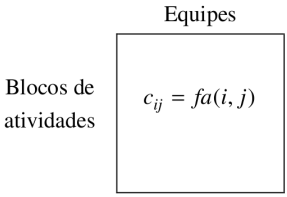

Figura 4. Estrutura da matriz de custo para o *k-swap* que se diferencia da estrutura da matriz do procedimento de construção da solução.

Considere ti o índice da hora iniciada mais cedo pelas escalas das equipes e tf o índice da hora finalizada mais tardiamente pelas escalas das equipes. A descrição do algoritmo do *k-swap* é dada pela Figura 5.

Procedimento *k-Swap*
Início
  $z^* = \infty$;
  Para $t = (ti + 1)$ **até** $(tf - k - 1)$ **faça**:
    Construir a matriz de custo $[c_{ij}]$ para o bloco $ba(k,t)$;
    Resolver o problema $PD([c_{ij}])$;
    Se $z < z^*$ **então**
      $z^* := z$;
      $t^* := t$;
  Associar os blocos para as equipes conforme a solução de $PD([c_{ij}])$ de t^*;
Fim.

Figura 5. Procedimento de melhoria *k-swap*.

A Figura 6 ilustra um exemplo de *1-swap* para $t = 4$ sobre um grafo representando as atividades a serem realizadas por 4 equipes num período de 7 horas, para o carregamento de 18 cargas. Os vértices rotulados representam as cargas, enquanto que os vértices vazios representam horários

sem carregamento (descanso ou viagem). As arestas contínuas representam a sequência de atividades previamente definidas, enquanto as arestas tracejadas são possibilidades de trocas que serão investigadas pelo *k-swap*.

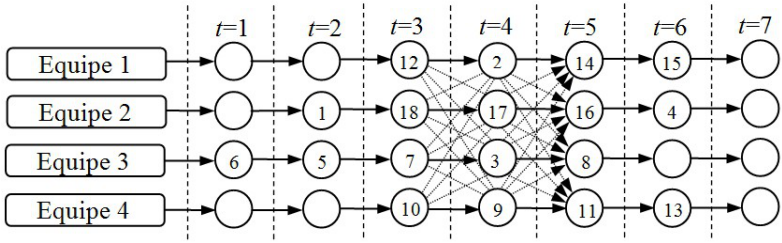

Figura 6. Ilustração de *1-swap* considerando somente um intervalo de tempo no horário $t = 4$.

A Figura 7 ilustra um exemplo de *2-swap* para os intervalos de tempo $h = 4$ e 5.

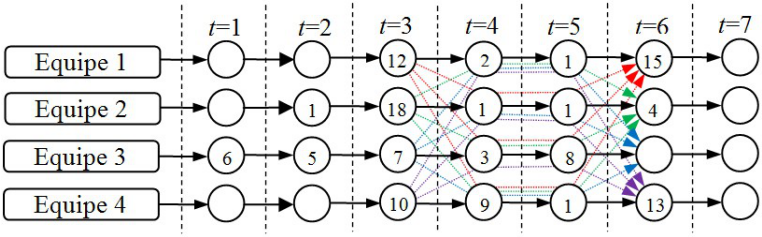

Figura 7. Ilustração de *2-swap* considerando dois intervalos de tempo consecutivos, $t = 4$ e $t = 5$.

O segundo procedimento de busca é denominado de PCR (Procedimento de Corte e Recombinação) em que são realizados "cortes" nas escalas entre dois horários de trabalho, separando as sequências de atividades em duas partes. As partes são recombinadas utilizando o PD. A Figura 8 ilustra o procedimento PCR para um corte.

A matriz de custo $[c_{ij}]$, neste caso, representa o custo de associar a sequência de atividades i antes do corte com a sequência j depois do corte. O algoritmo que descreve PCR é apresentado na Figura 9.

O algoritmo geral de busca local, denominado *bl-VND*, é um algoritmo baseado na meta-heurística *Variable Neighbourhood Descent* (VND), que explora o espaço de soluções por sistemáticas mudanças da estrutura de vizinhança (Hansen & Mladenović, 2001). Seja R o conjunto de vizinhanças,

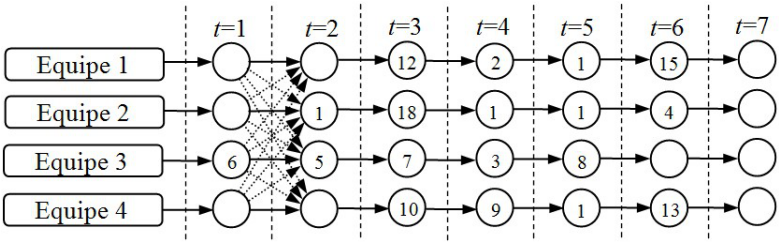

Figura 8. Ilustração do PCR sendo aplicado no corte entre o horário $t = 1$ e $t = 2$.

Procedimento PCR
Início
$z^* = \infty$;
Para $t = ti$ **até** $(tf - 1)$ **faça**:
Construir a matriz de custo $[c_{ij}]$para o corte entre t e t+1;
Resolver o problema $PD([c_{ij}])$;
Se z<z* **então**
$z^* := z$;
$t^* := t$;
Alocar as sequências de atividades conforme a solução de $PD([c_{ij}])$ de t^*;
Fim.

Figura 9. Procedimento de melhoria PCR.

$N_1, N_2, ..., N_R$. Se a solução corrente não é melhorada usando uma particular vizinhança, então uma próxima vizinhança é explorada e assim por diante. Para o algoritmo em questão foram utilizadas $R = 6$ vizinhanças, sendo $N_1 = 1\text{-}swap$, $N_2 = 2\text{-}swap$,..., $N_5 = 5\text{-}swap$ e N_6 =PCR.

Em cada iteração do *bl-VND* todas as vizinhanças são exploradas e o algoritmo para quando nenhuma melhoria ocorre em uma das iterações. A avaliação de cada solução é derivada da Equação 7 que pode ser descrita como:

$$f1(s) = \sum_{i=1}^{ne_{max}} fa(i,j) \mid j \in SC(i), \tag{8}$$

onde s é a solução com as escalas das equipes, $SC(i)$ é o conjunto de atividades da escala da equipe i e $fa(i,j)$ é o custo da atividades j estar na escala da equipe i. A estrutura geral do algoritmo é apresentada na Figura 10. Quando não ocorrer melhoras na solução corrente depois de

passar por todas as R vizinhanças, então o algoritmo para (critério de parada).

Procedimento *bl-VND*;
Inicialização: Selecione um conjunto de estrutura de vizinhança N_r (r = 1, ..., 6); selecione uma solução inicial s;
Início
Repetir até que o critério de parada seja satisfeito:
(1) $r \leftarrow 1$;
(2) **Repetir** até que $r = R$:
(a) Busca local: Encontrar uma solução s' de s ($s' \in N_r(s)$);
(b) **Se** $f1(s') < f1(s)$ **então** $s \leftarrow s'$ e $r \leftarrow 1$; **senão** $r \leftarrow r+1$;
Fim.

Figura 10. Procedimento geral de busca local utilizando todos os procedimentos de melhoria anteriores.

4.3 Busca local baseada em fila

Os procedimentos construtivos e de busca local apresentados têm como foco a minimização dos custos de horas pagas. Portanto, são avaliadas apenas as horas trabalhadas pelas equipes sem considerar o impacto no funcionamento de cada fábrica no que diz respeito ao tempo de espera das cargas no pátio das fábricas e tampouco a hora ociosa das fábricas. Com o intuito de considerar este custo, foi introduzida uma segunda função para avalizar uma solução definida como:

$$f2(s) = tq + po, \tag{9}$$

onde tq é o tempo médio na fila de espera das fábricas e po é uma penalidade caso ocorra alguma ociosidade das fábricas, que neste caso foi penalizada a ocorrência de ociosidade da fábrica.

Visando a minimização do tempo de espera no pátio de cada fábrica e a ociosidade das mesmas, foi projetado um segundo algoritmo de busca local, denominado *bl-Queue*, que realiza mudanças na hora de início das escalas de cada equipe. A hora de início da escala para cada equipe é adiantada ou atrasada de tal maneira que o custo $f2$ seja minimizado. Isto pode ser feito avaliando o impacto de cada escala iniciando no tempo $t, t = 1, ..., t_{max}$, sendo t_{max} o *tempo máximo* permitido para que a última carga da escala chegue na fábrica em tempo para descarregamento e abate. Este processo é repetido sequencialmente para cada equipe. O procedimento é apresentado na Figura 11.

Para calcular o valor de tq foi utilizada uma técnica de simulação discreta conhecida como método das 3-fases (Chwif & Medina, 2007). O

Procedimento *bl-Queue*
Início
Para *e*=1 **até** *ne* **faça**:
Encontrar um tempo *t* para início da escala da equipe *e* de tal maneira que *f*2 seja mínimo;
Mudar a hora de início da escala da equipe *e* para *t*;
Fim.

Figura 11. Procedimento de busca local baseado na informação da fila de espera na fábrica.

método considera dois tipos de eventos: condicional e programável. O evento programável é aquele que ocorre no tempo que estava sendo agendado para acontecer, mas o evento condicional é o tipo de evento que só ocorre quando as condições que ele requer forem satisfeitas. Todo evento condicional passa por uma fila de espera na qual permanece até que as condições sejam satisfeitas a fim de que seja executado (por exemplo, se a fábrica já iniciou o funcionamento ou se já é possível fazer o descarregamento de algum caminhão). Uma agenda é utilizada para colocar em ordem de ocorrência os eventos programáveis como também para atualizar o relógio da simulação. O método se resume nas três fases a seguir:

1. Checar o tempo de todos os eventos da agenda e selecionar o que ocorre primeiro. Atualizar o relógio da simulação para o tempo do evento selecionado.
2. O evento selecionado é executado e as entidades (cargas) são movidas para as respectivas filas de espera para a próxima atividade.
3. Pesquisar as entidades que estão na fila de espera e iniciar os eventos que satisfazem as condições do momento. Mover as entidades das filas para a atividade e calcular o tempo que a entidade permaneceu na fila.

Note-se que as próprias cargas, definidas anteriormente, podem ser interpretadas como as entidades que praticaram os eventos da simulação. Uma vez que se sabe a hora que as equipes iniciam suas escalas de trabalho, é possível saber qual a hora que cada carga chegará à fábrica de destino. Esta informação é utilizada para calcular o tempo que cada entidade (carga) permaneceu na fila de espera. Ao final da simulação se obtém as estatísticas das filas, como por exemplo, o tempo de permanência e o tamanho médio das filas.

4.4 O algoritmo geral

Nesta investigação foi proposto um algoritmo evolutivo, baseado em algoritmos genéticos, combinado com busca local. A abordagem por algoritmos

genéticos foi escolhida pela facilidade de tratar problemas complexos onde há dificuldades para a modelagem matemática e definição de um procedimento de busca de soluções. O problema investigado é altamente restrito e o uso de algoritmos genéticos tem sido indicado para estas situações (Podgorelec & Kokol, 1997).

O algoritmo é dividido em duas fases executadas consecutivamente, onde cada fase compreende um algoritmo genético. Porém, cada fase utiliza uma busca local diferente. Na fase 1 o algoritmo procura gerar as escalas de tal maneira que as horas pagas sejam minimizadas. Por outro lado, na fase 2 o algoritmo modifica a hora de início das escalas de cada equipe de tal maneira que o tempo médio na fila de espera da fábrica seja minimizado. Portanto, a primeira fase utiliza uma função de avaliação $f1$ (Equação 8) baseada em horas pagas, enquanto que a segunda fase utiliza $f1 + f2$ (Equações 8 e 9, respectivamente). Antes de iniciar a execução das fases 1 e 2, um modelo de programação linear inteira é resolvido para identificar a designação das cargas às fábricas minimizando custos de transporte e respeitando a capacidade de processamento de cada fábrica.

Cada indivíduo s (solução do problema) é codificado por um cromossomo que representa o conjunto de cargas, implementada em uma lista. O objetivo do algoritmo é encontrar a hora de início de carregamento de cada carga e atribuí-la a uma equipe.

A população inicial é gerada aleatoriamente com auxílio do procedimento Contrução apresentado na Seção 4.1. A população é atualizada com base na estratégia *steady state* (algumas vezes denominada de estacionária ou substituição incremental) para cada indivíduo gerado. O indivíduo substituirá aleatoriamente um membro da população com uma avaliação abaixo da média. Cada indivíduo gerado significa uma iteração do algoritmo. O número máximo de iterações é definido pelo parâmetro *MaxIt*. O uso da abordagem *steady state* tem sido utilizada para resolver muitos problemas de otimização, em particular os problemas de escalonamento (Beasley & Chu, 1996; Jat & Yang, 2009; AlSharafat & AlSharafat, 2010). Segundo Beasley & Chu (1996), um algoritmo genético que utiliza esta abordagem tende a convergir mais rápido do que o método geracional.

A seleção dos indivíduos para reprodução é realizada pela técnica do torneio, selecionando dois indivíduos aleatoriamente e escolhendo o mais apto com base na função de avaliação de cada fase.

O operador de cruzamento seleciona dois indivíduos e os recombina utilizando a técnica do cruzamento uniforme das cargas. As designações das cargas às equipes são realizadas com o procedimento de construção. Os indivíduos selecionados (pais) sempre geram um único indivíduo (filho). Nesta pesquisa não foi utilizado o parâmetro de taxa de cruzamento porque foi observado que o problema investigado é altamente restritivo. Isto significa que nem sempre o cruzamento gera uma solução viável (atendendo as restrições). Portanto, todas as soluções inviáveis são descartadas.

Foi observado nos experimentos que, dependendo da instância, a taxa de soluções inviáveis poderiam chegar até a 40% dos indivíduos gerados. O uso de um operador de cruzamento híbrido com o procedimento de construção tem sido uma alternativa para reduzir esses casos de inviabilidade em problemas altamente restritivos (Burke et al., 1995).

A mutação é aplicada segundo uma taxa a ser definida como parâmetro do algoritmo. A mutação realiza a mudança da hora de início de duas cargas selecionadas aleatoriamente. Este operador genético tem uma função muito importante para introduzir diversidade de soluções, uma vez que a população inicial é criada para horários iniciando em $t = 1$.

Na primeira fase do algoritmo os indivíduos são avaliados com base nos custos das horas pagas, enquanto que na segunda fase avalia-se o tamanho das filas nas fábricas utilizando uma técnica de simulação discreta. Assim, após cada reprodução e mutação é aplicado o algoritmo de busca local *bl-VND* na fase 1 e *bl-Queue* na fase 2.

O critério de parada é um parâmetro a ser definido em termos de número de iterações. Quando o critério de parada é alcançado o algoritmo devolve o melhor indivíduo da população. O número de iterações é utilizado como critério de parada para cada fase. Porém, a fase 2 utiliza apenas um percentual Ω do número de iterações da fase 1.

Cada fase do algoritmo segue os padrões clássicos dos algoritmos genéticos, mas a população é inicializada na fase 1, enquanto que a fase 2 continua operando sobre a população resultante da fase 1. A Figura 12 mostra uma visão geral do algoritmo proposto.

5. Resultados Computacionais

O algoritmo passou por várias experimentações piloto, as quais serviram para calibrar os parâmetros do próprio algoritmo e também para modificar algumas de suas características, de modo a chegar à versão final. Foram realizados extensivos testes computacionais e diferentes parâmetros foram testados de forma empírica. Para os resultados computacionais apresentados nesta seção, os parâmetros finais utilizados são descritos na Tabela 2.

Tabela 2. Parâmetros do algoritmo proposto

Parâmetro	**Valor**
Tamanho da população	60
Taxa de mutação	3%
Número de iterações – *MaxIt*	1000
Ω	50%
ph	100
po	10.000

Os experimentos foram realizados num computador com CPU Intel I3, 2,26 GHz, com Windows® 7 64 bits, usando a linguagem Object Pascal com

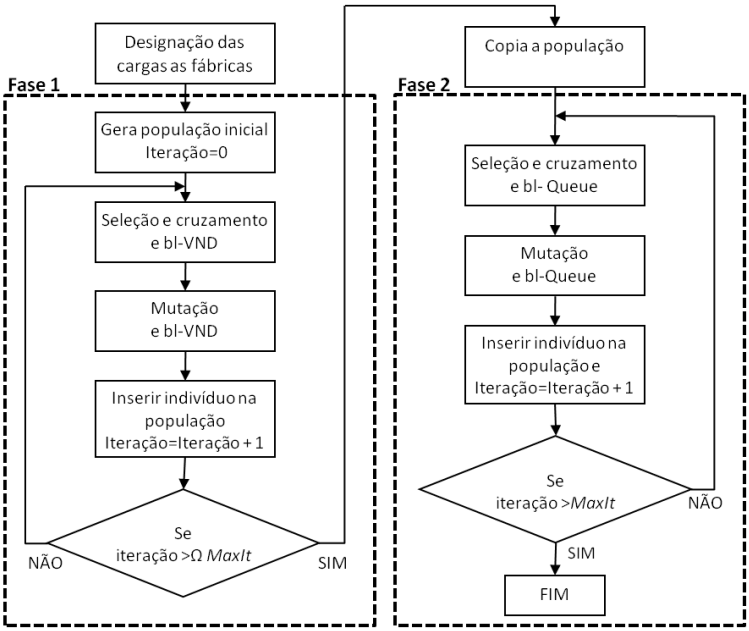

Figura 12. Fluxograma do algoritmo geral, incluindo as duas fases baseadas em algoritmos genéticos.

compilador FreePascal. Foram utilizadas escalas reais de uma empresa, obtidas ao longo de um período de 2 meses de operação, totalizando 45 dias. Envolveram um total de 112 catadores divididos em equipes de 14 pessoas (incluindo o motorista). A tabela 3 compara os valores obtidos pelo algoritmo com as escalas produzidas manualmente pela empresa (coluna Manual).

Os resultados mostram uma redução significativa dos custos operacionais. Os custos com horas pagas são relativos às equipes, o que significa que o custo reduzido deve ser multiplicado pelo número de pessoas envolvidas, neste caso 14 pessoas. A tabela também apresenta o número de escalas inviáveis que também contribui para os custos operacionais. Uma escala típica inviável planejada pela empresa ocorre quando o tempo real de deslocamento entre duas granjas não é contabilizado corretamente. Isto significa que certamente a escala não será executada dentro do tempo que havia sido planejado. Isto acarreta custos operacionais adicionais. Os resultados mostram uma redução de 100% destes casos. Do ponto de vista qualitativo foram observadas algumas eliminações de escalas indesejadas,

Tabela 3. Resultados computacionais com o algoritmo proposto.

	Manual	Algoritmo	Redução	% de redução
Total de horas pagas (horas)	3.290,60	2.938,18	352,42	10,71%
Total de distância percorrida (Km)	315.807,80	295.945,00	19.862,80	6,29%
Total de horas da fábrica parada (horas)	39,43	0,00	39,43	100,00%
Número médio de caminhões esperando no pátio	4,52	2,25	2,27	50,22%
Tempo médio de espera no pátio (minutos)	119,04	4,93	114,11	95,86%
Sobrecarga de caminhões no pátio *	64	0	64	100,00%
Número de escalas inviáveis	39	0	39	100,00%

*Quantidade de vezes que o limite de caminhões no pátio da fábrica foi ultrapassado.

como, por exemplo, escalas com extensos períodos de trabalho sem um intervalo de descanso.

O total de horas ociosas da fábrica foi reduzido para zero. O custo de uma fábrica parada é geralmente muito alto, podendo chegar ao valor de U$ 6.000,00 por hora parada.

Além de reduzir o número médio de caminhões no pátio da fábrica em aproximadamente 50%, o tempo médio de espera teve uma redução ainda mais acentuada, superior a 95%. Nota-se, ainda, uma redução de 100% dos casos de excesso de caminhões no pátio da fábrica. Nestes casos de excesso os caminhões ficam em um local impróprio, podendo comprometer a carga e aumentar os custos operacionais.

O tempo total de experimentação foi de aproximadamente 90 minutos para executar todas as instâncias do problema. Algumas instâncias consumiram aproximadamente 10 minutos para obtenção de uma solução.

6. Discussão e Conclusões

Neste trabalho foi investigado um algoritmo heurístico híbrido aplicado a um problema de escalonamento de atividades na agroindústria. Os ajustes no algoritmo e calibração dos parâmetros foram resultados de extensivas execuções piloto com dados de casos reais.

Os experimentos foram realizados com dados reais de uma empresa que trabalha com exportação de derivados de aves. Em média, o método possibilitou a redução aproximada dos seguintes aspectos: 11% dos custos

com horas pagas, 7% com custos de transporte, mais de 90% do tempo de espera no pátio do abatedouro e ainda eliminou os casos de ociosidade da fábrica. Além disto, foram observadas outras melhorias nas escalas das equipes do ponto de vista qualitativo. Embora não seja um objetivo explícito, o número de equipes também é minimizado automaticamente uma vez que o algoritmo procura reduzir custos com horas pagas.

Do ponto de vista do estado da arte, este projeto proporcionou a investigação de novos procedimentos aplicados ao escalonamento de atividades extensíveis a outros problemas similares. Algoritmos genéticos tiveram um importante papel neste projeto pela sua flexibilidade e facilidade de trabalhar com a resolução de um problema de difícil modelagem matemática.

Do ponto de vista da aplicabilidade, os experimentos realizados com o algoritmo proposto demonstraram a viabilidade de sua utilização num sistema computacional para automatizar a construção de escala de trabalho em situações reais. Além da agilidade na geração das escalas comparadas com o processo manual, notou-se uma redução significativa em termos quantitativos (custos) e qualitativos. Na incorporação do algoritmo num sistema computacional voltado para um gerente operacional da empresa, possivelmente os valores dos parâmetros do algoritmo deva ficar em segundo plano ou até mesmo ocultos.

Como investigações futuras pretende-se utilizar a abordagem geracional, juntamente com elitismo e utilizar uma taxa de mutação variável, conforme proposto por Beasley & Chu (1996), que aplica uma taxa de mutação mais elevada quando a avaliação (*fitness*) do melhor indivíduo fica próxima do pior indivíduo da população. Pesquisar por novos procedimentos de busca local e novos operadores genéticos também podem ser novas propostas de investigações futuras.

Agradecimentos

Os autores agradecem à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro ao projeto.

Referências

AlSharafat, W. & AlSharafat, M., Adaptive steady state genetic algorithm for scheduling university exams. In: *Proceedings of International Conference on Networking and Information Technology (ICNIT)*. p. 70–74, 2010.

Atlason, J.; Epelman, M. & Henderson, S., Optimizing call center staffing using simulation and analytic center cutting-plane methods. *Management Science*, 54(2):295–309, 2008.

Bansal, S.; Gowtham, K. & Hota, C., Novel adaptive scheduling algorithm for computational grid. In: *Proceedings of the 3rd IEEE international*

Conference on Internet Multimedia Services Architecture and Applications. p. 308–312, 2009.

Beasley, J. & Chu, P., A genetic algorithm for the set covering problem. *European Journal of Operational Research*, 94(2):392–404, 1996.

Blazewicz, J.; Ecker, K.; Schmidt, G. & Weglarz, J., *Scheduling in Computer and Manufacturing Systems*. 2a edição. New York, USA: Springer-Verlag, 1994.

Bodin, L., Solving large vehicle routing and scheduling problems in small core. In: *Proceedings of the 1983 Annual Conference on Computers: Extending the Human Resource*. p. 27–37, 1983.

Burke, E.; Elliman, D. & Weare, R., A hybrid genetic algorithm for highly constrained timetabling problems. In: *Proceedings of the 6^{th} International Conference on Genetic Algorithms*. p. 605–610, 1995.

Chwif, L. & Medina, A., *Modelagem e Simulação de Eventos Discretos: Teoria e Aplicações*. 3a edição. São Paulo, SP: Edição dos Autores, 2007.

Contreras, A.; Valero, C. & Pinninghoff, J., Applying genetic algorithms for production scheduling and resource allocation. special case: A small size manufacturing company. In: Ali, M. & Esposito, F. (Eds.), Innovations in Applied Artificial Intelligence. Berlin, Germany: Springer-Verlag, v. 3533 de *Lecture Notes in Computer Science*, p. 55–70, 2005.

Hansen, P. & Mladenović, N., Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449–467, 2001.

Harrison, J. & Zeevi, A., A method for staffing large call centers based on stochastic fluid models. *Manufacturing & Service Operations Management*, 7(1):20–36, 2005.

Hart, E.; Ross, P. & Nelson, J., Scheduling chicken catching – an investigation into the success of a genetic algorithm on a real-world scheduling problem. *Annals of Operations Research*, 92(1):363–380, 1999.

Jat, S. & Yang, S., A guided search genetic algorithm for the university course timetabling problem. In: *Proceedings of the 4^{th} Multidisciplinary International Scheduling Conference: Theory and Applications*. Dublin, Ireland, p. 180–191, 2009.

Kotecha, K.; Sanghani, G. & Gambhava, N., Genetic algorithm for airline crew scheduling problem using cost-based uniform crossover. In: Manandhar, S.; Austin, J.; Desai, U.; Oyanagi, Y. & Talukder, A. (Eds.), Applied Computing. Springer Berlin / Heidelberg, v. 3285 de *Lecture Notes in Computer Science*, p. 84–91, 2004.

Lin, C.; Lai, K. & Hung, S., Development of a workforce management system for a customer hotline service. *Computers and Operations Research*, 27(10):987–1004, 2000.

Park, Y., A hybrid genetic algorithm for the vehicle scheduling problem with due times and time deadlines. *International Journal of Production Economics*, 73(2):175–188, 2001.

Podgorelec, V. & Kokol, P., Genetic algorithm based system for patient scheduling in highly constrained situations. *Journal of Medical Systems*, 21(6):417–427, 1997.

Sheen, G.J. & Liao, L.W., Scheduling machine-dependent jobs to minimize lateness on machines with identical speed under availability constraints. *Computers and Operations Research*, 34(8):2266–2278, 2007.

Sonmez, O. & Gursoy, A., A novel economic-based scheduling heuristic for computational grids. *International Journal of High Performance Computing Applications*, 21(1):21–29, 2007.

Wren, A. & Wren, D., A genetic algorithm for public transport driver scheduling. *Computers and Operations Research*, 22(1):101–110, 1995.

Yunes, T.; Moura, A. & de Souza, C., Solving very large crew scheduling problems to optimality. In: *Proceedings of the 2000 ACM Symposium on Applied Computing*. New York, USA: ACM, v. 1, p. 446–451, 2000.

Zhang, F.; Cao, X. & Yang, D., Intelligent scheduling of public traffic vehicles based on a hybrid genetic algorithm. *Tsinghua Science & Technology*, 13(5):625–631, 2008.

Notas Biográficas

Ademir Aparecido Constantino é graduado em Matemática e tem mestrado e doutorado em Engenharia de Produção na área de Otimização e Simulação, além de Pós-doutorado na Univesidade de Nottingham, Inglaterra. Atualmente é professor titular do Departamento de Informática da Universidade Estadual de Maringá.

Dário Landa-Silva tem graduação e doutorado em Ciência da Computação. Atualmente é professor na Faculdade de Ciência da Computação na Universidade de Nottingham, Inglaterra.

Wesley Romão é graduado em Engenharia Química, tem mestrado em Ciência da Computação e doutorado em Engenharia de Produção. Atualmente é professor adjunto do departamento de Informática da Universidade Estadual de Maringá.