Roteamento de técnicos e programação de tarefas com janelas de tempo através do Algortimo Genético com Chaves Aleatórias Viezadas

Ricardo de Brito Damm, Débora Pretti Ronconi
Departamento de Engenharia de Produção da Escola Politécnica da USP
Av. Prof. Almeida Prado, 128, Cidade Universitária, 05508-070, São Paulo SP
rbdamm@usp.br, dronconi@usp.br

RESUMO

O problema de escalonamento de técnicos de campo consiste em programar um conjunto de tarefas que devem ser executadas por um grupo de técnicos. As tarefas estão espalhadas por uma cidade, têm diferentes prioridades e janelas de tempo. Os técnicos possuem habilidades distintas e diferentes horários de trabalho. Cada tarefa é atendida por um único técnico e, como objetivo, procura-se principalmente executar as tarefas mais prioritárias. O trabalho apresenta um modelo de programação linear inteira mista e, dada a complexidade do problema, foram desenvolvidas heurísticas construtivas e um Algoritmo Genético denominado *Biased Random Key Genetic Algorithm* (BRKGA), que utiliza chaves aleatórias para codificar e decodificar as soluções do problema. Testes computacionais foram realizados com 260 instâncias. Comparações com limitantes superiores e com soluções ótimas (instâncias de pequenas dimensões) indicaram um bom desempenho do BRKGA.

PALAVRAS CHAVE. Roteamento de técnicos, janelas de tempo, Biased Random Key Genetic Algorithm.

Área Principal: MH, L&T, OC

ABSTRACT

In the field technician scheduling problem a group of technicians have to execute a set of jobs or service tasks. The tasks are dispersed in a city, with different time windows, priorities, and processing times. Technicians have different skills and working hours. Each task is executed by only one technician and the main objective is to perform the priority tasks. The paper presents a mixed integer linear programming model and, due to the complexity of this problem, constructive heuristics and a customized Biased Random Key Genetic Algorithm (BRKGA), that uses random keys to code and decode the solutions. Computational tests with 260 instances are presented. Comparative study with optimal solutions (small-sized problems) and with the upper bounds validate the effectiveness of the proposed BRKGA.

KEYWORDS. Routing and scheduling technicians, time windows, Biased Random Key Genetic Algorithm

1. Introdução

O Problema de Escalonamento de Técnicos de Campo (PETC) é um tópico frequentemente encontrado em empresas prestadoras de serviço, principalmente em empresas de telecomunicações (Cordeau *et al.*, 2010; Hashimoto *et al.*, 2011; Kovacs *et al.*, 2011; Tsang e Voudouris, 1997) e serviços de manutenção (Cortés *et al.*, 2014; Overholts II *et al.*, 2009; Li *et al.*, 2005). O problema consiste em associar um número de tarefas (em diversos locais de uma região, com diferentes prioridades e janelas de tempo) a uma quantidade de técnicos (com diferentes horários de trabalho e habilidades distintas), que devem retornar para o local de origem no final do expediente. Cada tarefa é executada por um único técnico.

Entre os primeiros autores que estudaram o PETC estão Tsang e Voudouris (1997) e Xu e Chiu (2001). Em 2007, a Sociedade de Pesquisa Operacional da França (French Operational Society) e a France Telecom lançaram este problema como um desafio e os trabalhos de Cordeau *et al.* (2010) e Hashimoto *et al.* (2011) foram premiados. Em outra importante publicação, Kovacs *et al.* (2011) estudou uma extensão da pesquisa dos dois artigos anteriores. Recentemente, Pillac *et al.* (2012) analisaram a similaridade entre o PETC e o problema de roteamento de veículos com janelas de tempo (PRVJT). De fato, segundo Kovacs *et al.* (2011), Xu e Chiu (2001) e Pillac *et al.* (2012), o PETC é uma generalização do PRVJT, que é NP-difícil e, por isso, não conseguiria ser resolvido em tempo polinomial por métodos exatos.

O trabalho está estruturado da seguinte maneira: a Seção 2 apresenta um modelo de programação linear inteira mista (PLIM) e, dada a complexidade do problema, heurísticas construtivas foram criadas (Seção 3), um Algoritmo Genético foi aplicado ao problema (Seção 4) e modelos de limitantes superiores foram desenvolvidos (Seção 5). A Seção 6 apresenta os testes computacionais e a Seção 7, as principais conclusões.

2. Modelo de Programação Linear Inteira Mista

A seguir será apresentado o modelo de Programação Linear Inteira Mista (PLIM) para o Problema de Escalonamento de Técnicos de Campo (PETC), que está baseado na proposta de Xu e Chiu (2001). Seja $J = \{1, ..., n\}$ um conjunto de tarefas ou serviços e $K = \{1, ..., m\}$ um conjunto de técnicos disponíveis para executá-los. Cada tarefa $i \in J$ tem duração de tempo estimada em p_i, prioridade w_i e deve ser iniciada e concluída dentro de um intervalo de horário específico $[e_i, l_i]$. As tarefas estão distribuídas em uma região e c_{ij} é o tempo de deslocamento do local i para o local j, com $i, j \in J \cup \{0\}$, onde 0 representa a origem ou a sede da empresa.

Cada técnico $k \in K$ tem seu horário de trabalho diário definido por $[a_k, b_k]$. A habilidade de um técnico k executar uma tarefa i é dada por s_{ik}, que pode ser 0 ou 1, onde 1 indica ser plenamente apto e 0, incapaz de executá-la. As variáveis do modelo PLIM são: y_{ik} é uma variável binária que assume 1 se a tarefa i é designada ao técnico k e 0, caso contrário; x_{ijk} também é uma variável binária que assume 1 quando uma tarefa i precede outra tarefa j na rota do técnico k e 0, caso contrário; t_i é o momento do início da execução de uma tarefa i, e z_k é o tempo ocioso do técnico k após retornar para a origem.

A seguir encontra-se o modelo PLIM. A função objetivo (1) maximiza principalmente a soma das prioridades das tarefas executadas e, secundariamente, a soma dos tempos de disponibilidade dos técnicos após o retorno à base (no fim do dia). Desse modo, com o primeiro e o principal objetivo procurar-se-á executar as tarefas mais prioritárias e com o objetivo secundário, reduzir o tempo de deslocamento dos técnicos e o tempo de espera para o iniciar a execução das tarefas. Os denominadores têm duas funções: tornar adimensionais os valores somados e garantir que sempre será melhor executar uma tarefa, reduzindo a ociosidade de um técnico, e nunca o contrário. Concretamente, quando uma tarefa é associada a um técnico ($y_{ik} = 1$), a expressão $\frac{w_i y_{ik}}{MW}$ sempre será maior ou igual a um, enquanto que o termo ε garante que a expressão $\frac{z_k}{MZ}$ sempre será estritamente menor do que um.

$$\max \sum_{i \in J} \sum_{k \in K} \frac{w_i y_{ik}}{MW} + \sum_{k \in K} \frac{z_k}{MZ} \tag{1}$$

Sujeito a:

$$y_{ik} \leq s_{ik} \qquad k \in K, i \in J \tag{2}$$
$$\sum_{k \in K} y_{ik} \leq 1 \qquad i \in J \tag{3}$$
$$\sum_{j \in J \cup \{0\} \setminus i} x_{ijk} = \sum_{j \in J \cup \{0\} \setminus i} x_{jik} = y_{ik} \qquad k \in K, i \in J \tag{4}$$
$$\sum_{i \in J} x_{0ik} \leq 1 \qquad k \in K \tag{5}$$
$$e_i \leq t_i \leq l_i - p_i \sum_{k \in K} y_{ik} \qquad i \in J \tag{6}$$
$$t_i + p_i + c_{ij} \leq t_j + M(1 - x_{ijk}) \qquad k \in K, i \neq j \in J \tag{7}$$
$$a_k + c_{0j} \leq t_j + M(1 - x_{0jk}) \qquad k \in K, j \in J \tag{8}$$
$$t_i + p_i + c_{i0} \leq b_k - z_k + M(1 - x_{i0k}) \qquad k \in K, i \in J \tag{9}$$
$$0 \leq z_k \leq b_k - a_k \qquad k \in K \tag{10}$$
$$x_{ijk}, y_{ik} \in \{0, 1\} \qquad i, j \in J, k \in K \tag{11}$$
$$t_i, z_k \in \mathbf{R}_+ \qquad i \in J, k \in K \tag{12}$$

Onde:

$$MW = \min_{i \in J} w_i$$
$$MZ = \max_{k \in K} (b_k - a_k) + \varepsilon$$
ε é um número pequeno
$$M = \max \left(\max_{i \in J} l_i, \max_{k \in K} a_k \right) + \max_{i,j \in J \cup \{0\}} c_{ij}$$

As equações (2) e (3) garantem que somente técnicos capazes executarão as tarefas e que cada tarefa será alocada a, no máximo, um técnico. Quando uma tarefa i for designada a um técnico k ($y_{ik} = 1$), pela restrição (4) haverá uma única antecessora e uma única sucessora de i. Cada técnico deve deixar a origem, no máximo, uma vez, como assegura (5). Pela restrição (6), cada tarefa será executada dentro de sua janela de tempo. As restrições (7) e (8) definem, respectivamente, a relação entre o início do processamento de uma tarefa com a execução de sua antecessora ou com o início do horário de trabalho do técnico. O tempo disponível de cada técnico no fim do dia é limitado por (9). As restrições (7), (8) e (9) evitam o *subtour*.

3. Heurísticas Construtivas

Duas heurísticas construtivas são apresentadas a seguir. Resumidamente, pode-se dizer que as tarefas são ordenadas de acordo com algum critério e depois inseridas nas rotas dos técnicos.

3.1. Menor tempo de viagem (*Shortest travel time* - STT)

Esta heurística é formada por quatro passos:

1. *Ordene as tarefas:* ordene as tarefas em ordem decrescente de ρ, definido por:

$$\rho_i = \begin{cases} w_i - \frac{NST_i}{m+1} & \text{se } \frac{n}{m} < 10 \\ w_i + \frac{w_i}{p_i} + \frac{p_i}{l_i - e_i} & \text{caso contrário} \end{cases} \tag{13}$$

onde NST_i é o número de técnicos capazes de executar a tarefa i.
Note que, se a instância tem poucas tarefas por técnico (menos que 10), então elas são ordenadas pela prioridade (w_i); a segunda parte da expressão servirá como um critério de desempate quando duas ou mais tarefas tiverem a mesma prioridade. Já quando o número de tarefas por técnico for alto, elas são ordenadas pela prioridade somada à prioridade por tempo ($w_i + \frac{w_i}{p_i}$); em caso de empate, então o valor de $\frac{p_i}{l_i - e_i}$ definirá a ordem das tarefas.
Selecione a tarefa i melhor classificada.

2. *Avalie os técnicos candidatos:* para cada técnico k que seja capaz de executar a tarefa i (ou seja, $s_{ik} = 1$), avalie todas as posições possíveis para a nova tarefa na sua rota. Escolha a posição que resulte na solução factível com maior z_k.
3. *Escolha um técnico:* se não houver soluções factíveis no passo 2, siga para o 4. Caso contrário, selecione o técnico k com o menor tempo de deslocamento total (desde o início do dia até o retorno à empresa). Em caso de empate, o segundo critério de decisão será o técnico com maior z_k.
4. *Critério de parada:* se a tarefa i for a última classificada, fim do algoritmo. Caso contrário, selecione a próxima tarefa i classificada no primeiro passo e volte para o passo 2.

3.2. Técnico mais próximo (*Nearest technician* - NT)

Essa heurística baseia-se nos mesmos passos da heurística construtiva STT, com exceção do passo 3. Nesse passo, se existirem soluções factíveis, será selecionado o técnico k cujo local da última tarefa alocada é o mais próximo da nova tarefa i. Quando dois ou mais técnicos estão igualmente próximos da nova tarefa, então será escolhido o técnico k com a menor habilidade ponderada $\sum_{j \in A} s_{jk} w_j$ (onde A é o conjunto de tarefas que a heurística construtiva ainda não tentou inserir na rota dos técnicos). Dessa forma, procura-se preservar os técnicos mais capazes de realizar tarefas ainda não avaliadas. O último critério de desempate é o menor número de identificação do técnico.

4. Biased Random Key Genetic Algorithm (BRKGA)

Os Algoritmos Genéticos (AG) foram propostos por Holland em 1975 e são inspirados na teoria da evolução proposta por Darwin. Cada solução do problema é representada por uma solução (um indivíduo ou um cromossomo) dentro de uma população, que é sucedida por outra ao longo de gerações. A qualidade de uma solução é medida pela sua aptidão (geralmente, pela função objetivo), que influencia a contribuição de cada indivíduo para a geração seguinte e tende a guiar a busca do algoritmo em direção às regiões mais promissoras (Goldberg e Holland, 1988).

Procurando melhorar o desempenho do AG em problemas combinatórios, Bean (1994) propôs um AG com chaves aleatórias, denominado *Random-Key Genetic Algorithms* (RKGA). Trata-se de um AG que garante a factibilidade das novas soluções geradas ao longo da busca e explora a região factível indiretamente, através de uma busca no espaço das chaves aleatórias (Morán-Mirabal *et al.*, 2014). O RKGA foi utilizado em diversas pesquisas, apresentando bons resultados (por exemplo, Gonçaves *et al.*, 2005).

Alguns anos depois, novos conceitos foram introduzidos a esse AG, aumentando o protagonismo das melhores soluções (as mais aptas) de cada geração. Denominada por Gonçalves e Resende (2011) como *Biased Random Key Genetic Algorithm* (BRKGA), esse AG obteve bons resultados em diversas pesquisas: em problemas de máquina única (Valente e Gonçalves, 2009), problemas de cobertura (Resende *et al.*, 2012), dimensionamento de lotes (Gonçalves e Sousa, 2012), telecomunicações (Duarte *et al.*, 2014; Noronha *et al.*, 2011), atracamento de navios (Lalla-Ruiz *et al.*, 2014), coleta de doações de sangue (Grasas *et al.*, 2014), transportes (Buriol *et al.*, 2010) e o problema do caixeiro viajante (Morán-Mirabal *et al.*, 2014). Alguns estudos comparativos demonstraram um melhor desempenho do BRKGA em relação ao RKGA e ao AG (Gonçalves *et al.*, 2014; Gonçalves e Resende, 2011).

A seguir encontra-se uma breve descrição do BRKGA. A população inicial é formada por vetores de números reais aleatórios (chaves aleatórias) no intervalo [0; 1), que serão decodificados por um algoritmo para transformá-los em soluções factíveis do problema estudado. Soluções das heurísticas construtivas podem ser codificadas e inseridas na população inicial. A população de cada geração é dividida em dois grupos: um primeiro grupo menor com as melhores soluções (elite) e um segundo grupo maior com as demais soluções (não elite). Em seguida, a próxima geração é formada: todas as soluções de elite são copiadas, algumas novas soluções (mutantes) são geradas aleatoriamente (aumentando a diversidade da população) e o restante da população é formado por cruzamentos dos indivíduos da geração anterior. No cruzamento é utilizado o *crossover* parametrizado: dois cromossomos são selecionados, sendo um do grupo de elite e outro do grupo não elite. Para a escolha de cada gene do filho, é feito um sorteio onde a probabilidade de a solução de elite transmitir o seu gene é maior do que a da outra solução, de modo que as chaves aleatórias da solução elite tendem a predominar no novo cromossomo gerado. Não há mutações após o cruzamento e a diversificação da busca é garantida pelas soluções mutantes inseridas em cada geração. Uma vez formada uma nova população, todas as operações anteriores são repetidas, até um critério de parada ser atingido. A melhor solução da última geração será a solução do problema.

Um dos elementos mais relevantes do BRKGA é a definição do decodificador, que transformará os cromossomos em soluções factíveis para um problema. Além disso, é importante definir um codificador, que transforme as soluções das heurísticas construtivas em cromossomos codificados. A seguir, serão apresentadas três propostas de codificadores e decodificadores (ou três versões do BRKGA para o PETC): as duas primeiras baseiam nas heurísticas construtivas apresentadas anteriormente e a terceira, utiliza as chaves aleatórias para escolher técnicos para cada tarefa e construir suas rotas.

4.1. BRKGA-STT:

Esta versão do BRKGA representa cada solução como um vetor ($\boldsymbol{\lambda}$) composto por n elementos que associam uma chave aleatória a uma tarefa. A população inicial contém, além dos cromossomos gerados aleatoriamente, a solução encontrada pela heurística construtiva STT, que é codificada pelo seguinte valor em cada chave aleatória:

$$\lambda[i] = \frac{\rho_i}{\rho_{max}} \tag{14}$$

onde ρ_i é definido em (13), $\rho_{max} = \max\{\rho_i | i = 1, \ldots, n\} + \nu$ e ν é um número pequeno.

Para decodificar cada vetor, coloque em ordem decrescente as chaves aleatórias; a primeira tarefa classificada será avaliada em todos os técnicos possíveis e um técnico será selecionado, de acordo com os passos dois e três da heurística STT, respectivamente; esse procedimento será repetido para a segunda, terceira e demais tarefas, até a última tarefa classificada.

4.2. BRKGA-NT:

Assim com no BRKGA-STT, esta versão também contém cromossomos com n elementos que associam uma chave aleatória a uma tarefa. A população inicial também contém cromossomos gerados aleatoriamente e a solução encontrada pela heurística construtiva NT será codificada com o valor de cada chave aleatória definido por (14). Para decodificar cada cromossomo, as chaves aleatórias serão ordenadas de modo decrescente e as tarefas serão designadas aos técnicos da primeira classificada até a última, utilizando os passos dois e três da heurística NT.

4.3. BRKGA-A:

Nessa versão do BRKGA, cada cromossomo ($\boldsymbol{\lambda}$) contém n chaves aleatórias, que são utilizadas para selecionar um técnico para cada tarefa e depois construir a rota de cada técnico. As tarefas serão designadas para os técnicos da seguinte maneira: se há dois técnicos capazes de realizar uma tarefa e a chave aleatória for um número no intervalo [0;1/2), então ela será associada ao

primeiro técnico; caso seja um número entre [1/2; 1), será designada ao segundo. Um procedimento análogo é realizado quando uma tarefa tem 3 ou mais técnicos aptos. Uma vez distribuídas todas as tarefas, para construir a rota de cada técnico foi extraído um índice ψ de cada chave aleatória, que é a porcentagem de distância entre o valor da chave aleatória e o início do subintervalo: concretamente, se no exemplo anterior a chave aleatória tem valor 0,25, então ela está exatamente na metade do subintervalo [0;1/2) e tem índice de 50%. Desse modo, as tarefas de cada técnico são ordenadas em ordem crescente do índice ψ e, da primeira à última, serão introduzidas no roteiro do técnico.

Portanto, o técnico selecionado para cada tarefa será definido por:

$$TC_i = \lceil \lambda[i] \cdot NST_i \rceil \tag{15}$$

onde TC_i é o n-th técnico com habilidade para executar a tarefa i e NST_i é o número de técnicos capazes de executar a tarefa i.

Uma vez associadas as tarefas aos técnicos, o índice (ψ_i) será calculado:

$$\psi_i = \frac{\lambda[i] - \frac{TC_i - 1}{NST_i}}{\frac{1}{NST_i}} = NST_i \cdot \lambda[i] - TC_i + 1 \tag{16}$$

e as tarefas serão classificadas em ordem crescente do índice (ψ_i). Começando da primeira até a última tarefa, todas as posições possíveis na rota do técnico serão avaliadas e a posição que resultar na solução factível com maior ociosidade (z_k) será escolhida. Se não houver solução factível (por conflitos de janelas de tempo, etc.) para introduzir uma tarefa na rota, então ela não será realizada.

As soluções das heurísticas construtivas (STT e NT) serão inseridas na população inicial do BRKGA-A com o valor $\lambda[i] = \frac{TC_i - \frac{\rho_i}{\rho_{max}}}{NST_i}$ para as chaves aleatórias. Se essas heurísticas não associarem alguma tarefa a nenhum técnico, então um técnico com habilidade será sorteado e o valor da chave será $\lambda[i] = \frac{TC_i - \frac{\nu}{\rho_{max}}}{NST_i}$, onde ν é um número muito pequeno.

5. Limitantes Superiores

Como o PETC é um problema NP-difícil, espera-se encontrar soluções ótimas apenas para instâncias de pequenas dimensões. Para avaliar os resultados obtidos pelas heurísticas construtivas e das versões do BRKGA para os problemas maiores, convém obter limitantes superiores de cada problema.

A seguir são apresentadas duas propostas de como relaxar o modelo de Programação Linear Inteira Mista (PLIM) para obter limitantes superiores. A primeira trata-se de uma proposta apresentada no trabalho de Xu e Chiu (2001) que foi aperfeiçoada; já a segunda foi desenvolvida pelos autores deste trabalho.

5.1. Modelo 1: simplificando os conceitos de janela de tempo e tempo de viagem

Para simplificar a ideia de janela de tempo, um conjunto E foi definido, contendo pares de tarefas que não podem ser executadas pelo mesmo técnico:

$$E = \{(i,j) | i \neq j \in J, e_i + p_i + c_{ij} > l_j - p_j \text{ e } e_j + p_j + c_{ji} > l_i - p_i\} \tag{17}$$

Com relação ao tempo de viagem, para cada técnico foi calculado um tempo de viagem (ou de espera) prévio mínimo δ_{ik} em relação ao local de uma tarefa i.

$$\delta_{ik} = \min\{\tau_{jik} \,|\, 0 \leq j \leq n, j \neq i\} \tag{18}$$

Onde:

$$\tau_{jik} = \begin{cases} \max\{c_{ji}, e_i - l_j\}, & \text{if } s_{ik} = s_{jk} = 1 \text{ e } e_j + p_j + c_{ji} \leq l_i - p_i \\ b_k - a_k, & \text{caso contrário} \end{cases} \quad (19)$$

Para determinar o tempo de viagem mínimo para um técnico k retornar à base, um parâmetro ϕ_k foi definido da seguinte maneira:

$$\phi_k = \min\{c_{i0} \mid i \in J, s_{ik} = 1\} \quad (20)$$

A seguir encontra-se o modelo completo desta primeira proposta de limitante superior:

$$\max \sum_{i \in J} \sum_{k \in K} \frac{w_i y_{ik}}{MW} + \frac{1}{MZ} \cdot \left(\sum_{k \in K} (b_k - a_k) - \sum_{i \in J} \sum_{k \in K} (p_i + \delta_{ik}) y_{ik} \right) \quad (21)$$

Sujeito a:

$$y_{ik} \leq s_{ik} \qquad k \in K, i \in J \quad (22)$$

$$\sum_{k \in K} y_{ik} \leq 1 \qquad i \in J \quad (23)$$

$$\sum_{i \in J} (p_i + \delta_{ik}) y_{ik} + \phi_k \leq b_k - a_k \qquad k \in K \quad (24)$$

$$y_{ik} + y_{jk} \leq 1 \qquad (i, j) \in E, k \in K \quad (25)$$

$$y_{ik} \in \{0, 1\} \qquad i \in J, k \in K \quad (26)$$

A função objetivo (21) é similar ao modelo PLIM, sendo que o objetivo secundário original (soma das ociosidades dos técnicos) foi substituído pelo número total de horas trabalhadas pelos técnicos menos os tempos de processamento (p_i) e os tempos prévios (δ_{ik}) das tarefas executadas. A restrição (22) garante que todas as tarefas são executadas por técnicos capazes e a restrição (23) determina que uma tarefa não pode ser associada a mais de um técnico. Pela restrição (24), a soma dos tempos de processamento, dos tempos prévios de deslocamento e do tempo mínimo para retornar à base não pode superar o número de horas de trabalho de cada técnico. Se duas tarefas são incompatíveis devido à janela de tempo, então elas pertencem ao conjunto E e a restrição (25) garante que não serão associadas ao mesmo técnico.

5.2. Modelo 2: Limitando o tempo máximo de trabalho dos técnicos e excluindo combinações infactíveis de tarefas

Neste modelo, em vez das restrições (24) e (25), outra abordagem será adotada para simplificar os tempos de viagem e as janelas de tempo. Primeiro, para cada técnico k foi feita uma enumeração completa de todas as tarefas possíveis ($s_{ik} = 1$) e determinado a maior soma dos tempos de processamento $(pmax_k)$ que ele pode executar, considerando os dados originais do problema (tempos de deslocamento, janelas de tempo do técnico e das tarefas). Analogamente, a maior soma ponderada dos tempos de processamento $(w_i p_i)$ foi calculada para cada técnico k $(pwmax_k)$. Para evitar soluções que excedam esses limites, as restrições (30) e (31) foram acrescentadas ao modelo.

Para evitar soluções infactíveis do modelo PLIM original, em vez do conjunto E, novas restrições foram incluídas. Para cada técnico k, considerando as tarefas possíveis ($s_{ik} = 1$), todos os grupos de até quatro tarefas foram formados. Para cada um desses grupos, foi feita uma enumeração completa e avaliado se existe alguma solução factível: caso não haja (por restrições de janela de tempo, por exemplo), as restrições (32)-(35) serão adicionadas no modelo para excluir esse grupo.

A seguir encontra-se o segundo modelo de limitante superior:

$$\max \sum_{i\in J}\sum_{k\in K}\frac{w_i y_{ik}}{MW} + \frac{1}{MZ}\cdot\left(\sum_{k\in K}(b_k - a_k) - \sum_{i\in J}\sum_{k\in K}(p_i + \delta_{ik})y_{ik}\right) \tag{27}$$

Sujeito a:

$$y_{ik} \leq s_{ik} \qquad k \in K, i \in J \tag{28}$$
$$\sum_{k\in K} y_{ik} \leq 1 \qquad i \in J \tag{29}$$
$$\sum_{i\in J} p_i y_{ik} \leq pmax_k \qquad k \in K \tag{30}$$
$$\sum_{i\in J} w_i p_i y_{ik} \leq pwmax_k \qquad k \in K \tag{31}$$
$$y_{ik} \leq 0 \qquad (i,k) \in A_1 \tag{32}$$
$$y_{i_1k} + y_{i_2k} \leq 1 \qquad (i_1,i_2,k) \in A_2 \tag{33}$$
$$y_{i_1k} + y_{i_2k} + y_{i_3k} \leq 2 \qquad (i_1,i_2,i_3,k) \in A_3 \tag{34}$$
$$y_{i_1k} + y_{i_2k} + y_{i_3k} + y_{i_4k} \leq 3 \qquad (i_1,i_2,i_3,i_4,k) \in A_4 \tag{35}$$
$$y_{ik} \in \{0,1\} \qquad i \in J, k \in K \tag{36}$$

Onde:
$A_1 = \{(i,k)|i \in J, k \in K, s_{ik} = 1, a_k + c_{0i} + p_i > l_i$ ou $e_i + p_i + c_{i0} > b_k\}$

$A_2 = \{(i_1,i_2,k)|i_1,i_2 \in J, k \in K, s_{i_1k}, s_{i_2k} = 1, (i_1,k),(i_2,k) \notin A_1,$ e todas as possíveis sequências de tarefas i_1 e i_2 são infactíveis para o técnico $k\}$

$A_3 = \{(i_1,i_2,i_3,k)|i_1,i_2,i_3 \in J, k \in K, s_{i_1k}, s_{i_2k}, s_{i_3k} = 1, (i_1,k),(i_2,k),(i_3,k) \notin A_1, (i_1,i_2,k),$ $(i_1,i_3,k),(i_2,i_3,k) \notin A_2$ e todas as possíveis sequências de tarefas i_1, i_2 e i_3 são infactíveis para o técnico $k\}$

$A_4 = \{(i_1,i_2,i_3,i_4,k)|i_1,...,i_4 \in J, k \in K, s_{i_1k},...,s_{i_4k} = 1, (i_1,k),...,(i_4,k) \notin A_1,$ $(i_1,i_2,k),...,(i_3,i_4,k) \notin A_2, (i_1,i_2,i_3,k),...,(i_2,i_3,i_4,k) \notin A_3$ e todas as possíveis sequências de tarefas i_1, i_2, i_3 e i_4 são infactíveis para o técnico $k\}$

6. Experimentos computacionais

Esta seção descreve as instâncias utilizadas e apresenta os resultados obtidos pelos métodos apresentados anteriormente. Todos os códigos foram escritos em linguagem C e os testes realizados em um computador com processador Intel 2.93 GHz com 16 GB de memória RAM.

6.1. Instâncias

Foram geradas 260 instâncias, com a configuração de parâmetros descrita a seguir. Quatro diferentes distribuições geográficas de locais das tarefas foram adotadas: distribuição uniforme (R), cluster (C), semi-cluster (RC) e radial (RAD); as três primeiras foram propostas por Solomon (1987) e a última trata-se de uma nova distribuição proposta neste trabalho, onde há uma maior densidade de tarefas na área central de uma cidade, que vai diminuindo gradativamente em direção à periferia. Para cada distribuição geográfica, 5 instâncias foram geradas para 13 diferentes casos (ou dimensões) de tarefas e técnicos, apresentados na Tabela 1. Todas as tarefas e técnicos têm janelas de tempo.

Os parâmetros das instâncias foram gerados com uma distribuição uniforme discreta nos seguintes intervalos: prioridade das tarefas (w_i) entre 1 (baixa) e 10 (alta); tempo de processamento

Tabela 1: Dimensão das instâncias geradas. Critério de parada (tempo máximo) do BRKGA ($limit\_time$).

Caso	# tarefas	# técnicos	$limit\_time$
1	16	2	3
2	26	2	5
3	30	3	10
4	39	3	50
5	45	7	75
6	64	5	80
7	80	13	180
8	100	10	250
9	120	20	480
10	150	25	600
11	200	33	900
12	500	83	1800
13	999	166	3600

das tarefas (p_i) entre 30 e 120 minutos; início da janela de tempo das tarefas (e_i) entre 7 e 19 horas; tamanho da janela de tempo das tarefas entre 2 e 8 horas; início da janela de tempo dos técnicos (a_k) entre 7 e 12 horas; fim da janela de tempo dos técnicos $b_k = a_k + 9$ horas; habilidade do técnico k para executar a tarefa i (s_{ik}) será 0 ou 1; foi considerado que todas as tarefas estão localizadas em uma região onde o tempo de deslocamento entre quaisquer dois locais é menor ou igual a 1,5 horas. Todos os parâmetros das instâncias foram obtidos por um gerador de números aleatórios proposto por Taillard (1993).

6.2. Heurísticas construtivas

Os resultados das heurísticas construtivas foram analisados por duas medidas de desempenho. A primeira é a Relação de Desempenho Individual (RDI) (Kim, 1993), dado por $RDI = \frac{B-H}{B-W}$, onde H é o valor da função objetivo obtido pela heurística construtiva e B e W são, respectivamente, o melhor e o pior valor encontrado entre todos os métodos avaliados. Note que quanto menor o valor do RDI, melhor é o desempenho da heurística. A segunda medida é o Número de Instâncias com Melhor Resultado ($NIMR$), i.e., o total de vezes que o melhor resultado (B) foi encontrado por cada heurística.

As duas heurísticas desenvolvidas neste trabalho foram comparadas com a heurística proposta por Xu e Chiu (2001), que será chamada de XC. A heurística XC é similar à NT, com duas diferenças: no primeiro passo a XC ordena as tarefas pela razão prioridade por tempo de processamento $(\frac{w_i}{p_i})$ e no terceiro passo não há o segundo critério de decisão.

Os valores médios obtidos pelas medidas $RDI(NIMR)$ foram: 0,28 (145) para STT, 0,36 (94) para NT e 0,77 (34) para XC. Portanto, as duas heurísticas propostas superaram o desempenho da heurística XC, com a heurística STT apresentando a melhor RDI média e encontrando 56% dos melhores resultados. Além disso, a heurística STT e NT melhoraram em média os resultados da heurística XC em 2,0% e 1,6%, respectivamente. A diferença percentual entre o limitante superior (UB) e o valor obtido pelas heurísticas (V) foi calculado por $GAP_{UB} = 100 \cdot \frac{UB-V}{UB}$ e encontra-se na Tabela 2.

6.3. BRKGA

6.3.1. Calibragem do BRKGA

Inicialmente os parâmetros do BRKGA foram calibrados a partir dos valores sugeridos por Gonçalves e Resende (2011). Concretamente: conjunto elite (n_e) igual a 15%, 20% ou 25% da população; novas soluções mutantes (n_m) igual a 5%, 10% ou 15% da população; probabilidade

de *crossover* parametrizado (p_e) igual a 50%, 60%, 70% ou 80%. O tamanho da população (n_p) máximo adotado foi de $15n$ cromossomos. Após a calibragem, os seguintes parâmetros foram fixados para cada uma das versões do BRKGA: n_e = 15%, n_m = 15% e p_e = 60% para o BRKGA-STT; n_e = 25%, n_m = 15% e p_e = 60% para o BRKGA-NT; n_e = 15%, n_m = 10% e p_e = 70% para o BRKGA-A. Com relação ao tamanho da população, foram adotados os seguintes valores para todas as versões do BRKGA: $15n$ nos casos 1 a 8; $10n$ no caso 9; $5n$ nos casos 10 e 11; n no caso 12 e $0,2n$ no caso 13.

6.3.2. Desempenho do BRKGA

Para comparar as diferentes propostas de BRKGA, foi adotado como critério de parada um tempo máximo $(limit\_time)$ suficiente para todas as três versões atingirem a convergência; o valor adotado em cada caso encontra-se na Tabela 1. Cada versão do BRKGA resolveu cinco vezes cada instância, a partir de diferentes sementes (Taillard, 1993). De acordo com a medida de desempenho RDI, o BRKGA-STT ($RDI = 0,13$) e o BRKGA-NT ($RDI = 0,13$) apresentaram resultados médios equivalentes e foram superiores ao BRKGA-A ($RDI = 0,52$). Os BRKGA-STT e BRKGA-NT melhoraram os resultados obtidos pela heurística construtiva STT em média 6,6%, enquanto que a melhoria média do BRKGA-A foi de 6,0%.

Um estudo comparativo com as soluções ótimas para problemas com até 39 tarefas e 3 técnicos também foi realizado. Usando o software ILOG CPLEX, versão 12.6, com tempo de execução máximo de dez horas (iniciando a busca a partir de soluções encontradas pelo BRKGA) e enumerações completas, o valor ótimo foi obtido para as 20 instâncias dos casos 1 e 2, 19 instâncias do caso 3 e 6 instâncias do caso 4, totalizando 65 instâncias. O BRKGA-STT encontrou a solução ótima 325 vezes (100%), o BRKGA-NT 323 vezes (99%) e o BRKGA-A 319 vezes (98%). Nos casos em que o valor ótimo não foi encontrado, a diferença percentual média entre a solução encontrada e o valor ótimo foi de 0,5% (BRKGA-NT) e 0,8% (BRKGA-A).

Para os casos de maiores dimensões, a qualidade das soluções obtidas pelas heurísticas será avaliada através de uma comparação com o melhor limitante superior obtidos pelos modelos apresentados anteriormente. A diferença percentual entre esse limitante superior (UB) e o valor obtido pela heurística (V) foi calculado por $GAP_{UB} = 100 \cdot \frac{UB-V}{UB}$. A Tabela 2 apresenta a média dos melhores (mínimo) e dos piores (máximo) GAP_{UB} e a média geral para cada caso. Os desempenhos médios do BRKGA-STT e do BRKGA-NT foram semelhantes, com GAP_{UB} médio de 4,2% e 4,1% respectivamente para os casos 5 a 13, o que sugere um bom desempenho para o BRKGA, uma vez que o GAP_{UB} médio para as instâncias pequenas foi de 6,6%. Note que nos

Tabela 2: GAP_{UB} entre as heurísticas e os limitantes superiores. Para cada método, são apresentados: média dos melhores resultados para cada instância (Min), média geral (Média) e média dos piores valores encontrados para cada instância (Max). Os números em negrito indicam as melhores médias obtidas.

	STT	NT	XC	BRKGA-STT			BRKGA-NT			BRKGA-A		
Caso	Média	Média	Média	Min	Média	Max	Min	Média	Max	Min	Média	Max
1	7,9	8,9	9,8	**3,3**	**3,3**	**3,3**	**3,3**	**3,3**	**3,3**	**3,3**	**3,3**	**3,3**
2	14,3	15,7	16,1	**7,7**	**7,7**	**7,7**	**7,7**	**7,7**	7,8	**7,7**	**7,7**	**7,7**
3	13,2	14,7	15,8	**6,4**	**6,4**	**6,4**	**6,4**	**6,4**	**6,4**	**6,4**	6,5	6,5
4	16,4	16,2	19,5	**8,8**	**8,8**	**8,8**	**8,8**	**8,8**	**8,8**	**8,8**	8,9	8,9
5	10,1	9,5	12,4	**3,5**	**3,6**	**3,7**	3,6	**3,6**	**3,7**	3,7	4,0	4,4
6	19,3	20,1	20,5	**10,6**	**10,7**	11,0	**10,6**	10,8	**10,9**	**10,6**	10,9	11,4
7	10,0	10,0	11,3	**3,2**	3,4	**3,6**	**3,2**	**3,3**	**3,6**	3,7	4,1	4,4
8	18,7	18,0	19,8	**9,0**	**9,3**	9,7	9,1	**9,3**	**9,6**	9,7	10,2	10,8
9	7,8	7,7	9,2	2,1	2,3	2,4	**2,0**	**2,1**	**2,3**	3,0	3,2	3,5
10	7,3	7,3	8,7	2,1	2,3	2,4	**1,9**	**2,0**	**2,1**	3,1	3,5	3,8
11	6,4	6,6	7,7	2,0	2,2	2,5	**1,8**	**1,9**	**2,1**	3,1	3,3	3,6
12	3,8	4,2	5,1	**2,0**	**2,1**	**2,2**	**2,0**	**2,1**	**2,2**	2,7	2,9	3,0
13	2,7	3,5	3,9	**1,6**	**1,7**	**1,9**	2,0	2,1	2,3	2,2	2,3	2,5
Média (1-4)	13,0	13,9	15,3	**6,6**	**6,6**	**6,6**	**6,6**	**6,6**	**6,6**	**6,6**	**6,6**	**6,6**
Média (5-13)	9,6	9,7	11,0	**4,0**	4,2	4,4	**4,0**	**4,1**	**4,3**	4,6	4,9	5,3

casos em que há uma alta proporção de tarefas por técnicos (casos 6 e 8) o GAP_{UB} médio é 10%, enquanto que nos outros casos (5, 7, 9, 10, 11, 12 e 13) o GAP_{UB} médio é de 2,5%. Provavelmente essa diferença significativa deve-se à perda qualidade dos limitantes superiores quando crescem as proporções de tarefas por técnico, fato também observado nas instâncias pequenas: com menos de 10 tarefas por técnicos os resultados ótimos estão em média a 3,3% dos limitantes superiores, enquanto que essa proporção sobe para 7,6% nas instâncias com 10 ou mais tarefas por técnico.

Pode-se também ressaltar que a heurística construtiva STT obteve um bom desempenho para as instâncias com 500 e 999 tarefas, com GAP_{UB} médio de 3,8 e 2,7% respectivamente, em um tempo computacional inferior a 0,15 segundos. Além disso, esses resultados foram melhorados em média 1,7 e 1,0% respectivamente pelo BRKGA-STT.

7. Conclusão

Neste trabalho duas heurísticas construtivas e três versões do BRKGA foram desenvolvidas para o Problema de Escalonamento de Técnico de Campo (PETC). Para grandes instâncias (com 500 e 999 tarefas), a heurística construtiva STT obteve bons resultados em um tempo computacional reduzido. Isso pode ser atribuído por essa heurística explorar diversas características do problema (prioridades das tarefas, tempo de deslocamento e habilidade dos técnicos), sinalizando o potencial dessas heurísticas para problemas de grandes dimensões (Ronconi, 2004).

Duas versões do Algoritmo Genético com chaves aleatórias viezadas (BRKGA-STT e BRKGA-NT) encontraram 100% e 99% das soluções ótimas dos problemas de pequenas dimensões, respectivamente, e resultados médios inferiores a 4,2% dos limitantes superiores. Para problemas de médio porte (120 a 200 tarefas), o BRKGA-NT obteve melhor desempenho; já para grandes problemas (999 tarefas), o BRKGA-STT atingiu melhores resultados. O fato de essas versões terem superado o BRKGA-A sinaliza que decodificadores mais elaborados podem apresentar melhores resultados.

Agradecimentos

Este trabalho foi financiado pela FAPESP (2010/10133-0 e 2013/07375-0) e pelo CNPq.

Referências

Bean, J. C. (1994), Genetic algorithms and random keys for sequencing and optimization, *ORSA Journal on Computing*, 6(2), 154-160.

Buriol, L. S., Hirsch, M. J., Pardalos, P. M., Querido, T., Resende, M. G. e Ritt, M. (2010), A biased random-key genetic algorithm for road congestion minimization, *Optimization Letters*, 4(4), 619-633.

Cordeau, J. F., Laporte, G., Pasin, F. e Ropke, S. (2010), Scheduling technicians and tasks in a telecommunications company, *Journal of Scheduling*, 13(4), 393-409.

Cortés, C. E., Gendreau, M., Rousseau, L. M., Souyris, S. e Weintraub, A. (2014), Branch-and-price and constraint programming for solving a real-life technician dispatching problem, *European Journal of Operational Research*, 238(1), 300-312.

Duarte, A., Martí, R., Resende, M. G. C. e Silva, R. M. A. (2014), Improved heuristics for the regenerator location problem, *International Transactions in Operational Research*, 21(4), 541-558.

Gonçalves, J. F., de Magalhães Mendes, J. J. e Resende, M. G. (2005), A hybrid genetic algorithm for the job shop scheduling problem, *European Journal of Operational Research*, 167(1), 77-95.

Gonçalves, J. F. e Resende, M. G. (2011), Biased random-key genetic algorithms for combinatorial optimization, *Journal of Heuristics*, 17(5), 487-525.

Gonçalves, J. F., Resende, M. G. e Toso, R. F. (2014), An experimental comparison of biased and unbiased random-key genetic algorithms, *Pesquisa Operacional*, 34(2), 143-164.

Gonçalves, J. F. e Sousa, P. S. (2011), A genetic algorithm for lot sizing and scheduling under capacity constraints and allowing backorders, *International Journal of Production Research*, 49(9), 2683-2703.

Goldberg, D. E. e Holland, J. H. (1988), Genetic algorithms and machine learning, *Machine Learning*, 3(2), 95-99.

Grasas, A., Ramalhinho, H., Pessoa, L. S., Resende, M. G., Caballé, I. e Barba, N. (2014), On the improvement of blood sample collection at clinical laboratories, *BMC health services research*, 14(1), 12.

Hashimoto, H., Boussier, S., Vasquez, M. e Wilbaut, C. (2011), A GRASP-based approach for technicians and interventions scheduling for telecommunications, *Annals of Operations Research*, 183(1), 143-161.

Kim, Y. D. (1993), Heuristics for flowshop scheduling problems minimizing mean tardiness, *Journal of the Operational Research Society*, 19-28.

Kovacs, A. A., Parragh, S. N., Doerner, K. F. e Hartl, R. F. (2012), Adaptive large neighborhood search for service technician routing and scheduling problems, *Journal of Scheduling*, 15(5), 579-600.

Lalla-Ruiz, E., González-Velarde, J. L., Melián-Batista, B. e Moreno-Vega, J. M. (2014), Biased random key genetic algorithm for the tactical berth allocation problem, *Applied Soft Computing*, 22, 60-76.

Li, Y., Lim, A. e Rodrigues, B. (2005), Manpower allocation with time windows and job-teaming constraints, *Naval Research Logistics (NRL)*, 54 (4), 302-311.

Morán-Mirabal, L. F., González-Velarde, J. L. e Resende, M. G. C. (2014), Randomized heuristics for the family traveling salesperson problem, *International Transactions in Operational Research*, 21(1), 41-57.

Noronha, T. F., Resende, M. G. e Ribeiro, C. C. (2011), A biased random-key genetic algorithm for routing and wavelength assignment, *Journal of Global Optimization*, 50(3), 503-518.

Overholts II, D. L., Bell, J. E. e Arostegui, M. A. (2009), A location analysis approach for military maintenance scheduling with geographically dispersed service areas, *Omega*, 37(4), 838-852.

Pillac, V., Gueret, C. e Medaglia, A. L. (2013), A parallel matheuristic for the technician routing and scheduling problem, *Optimization Letters*, 7(7), 1525-1535.

Resende, M. G., Toso, R. F., Gonçalves, J. F. e Silva, R. M. (2012), A biased random-key genetic algorithm for the Steiner triple covering problem, *Optimization Letters*, 6(4), 605-619.

Ronconi, D. P. (2004), A note on constructive heuristics for the flowshop problem with blocking, *International Journal of Production Economics*, 87(1), 39-48.

Solomon, M. M. (1987), Algorithms for the vehicle routing and scheduling problems with time window constraints, *Operations Research*, 35(2), 254-265.

Taillard, E. (1993), Benchmarks for basic scheduling problems, *European Journal of Operational Research*, 64(2), 278-285.

Tsang, E. e Voudouris, C. (1997), Fast local search and guided local search and their application to British Telecom's workforce scheduling problem, *Operations Research Letters*, 20(3), 119-127.

Valente, J. M. e Gonçalves, J. F. (2009), A genetic algorithm approach for the single machine scheduling problem with linear earliness and quadratic tardiness penalties, *Computers & Operations Research*, 36(10), 2707-2715.

Xu, J. e Chiu, S. Y. (2001), Effective heuristic procedures for a field technician scheduling problem, *Journal of Heuristics*, 7(5), 495-509.