O USO DA HEURÍSTICA *ADAPTIVE LARGE NEIGHBORHOOD SEARCH* PARA RESOLVER O PROBLEMA DE PROGRAMAÇÃO DE TRIPULAÇÕES

Leandro do Carmo Martins
Universidade Federal de Ouro Preto – Departamento de Computação
Campus Morro do Cruzeiro, Ouro Preto, MG – Brasil - 35400-000
leandro.cm@live.com

Gustavo Peixoto Silva
Universidade Federal de Ouro Preto – Departamento de Computação
Campus Morro do Cruzeiro, Ouro Preto, MG – Brasil - 35400-000
gustavo@iceb.ufop.br

RESUMO

O Problema de Programação de Tripulações (PPT) consiste em minimizar o custo total com as tripulações que conduzirão uma frota de uma empresa de transporte público por ônibus urbano. Neste trabalho é apresentada a resolução do PPT por meio da heurística *Adaptive Large Neighborhood Search* (ALNS), que é caracterizada pela destruição e reparo da solução, através de heurísticas de remoção e inserção. A escolha dos pares de heurísticas utilizadas em cada iteração se baseia em pesos ajustados dinamicamente em função de suas eficiências ao longo da busca. A implementação foi testada com dados reais de uma empresa que opera em Belo Horizonte-MG, e gerou soluções superiores às adotadas pela empresa e também superiores às soluções geradas por outros métodos da literatura.

PALAVARAS-CHAVE. Problema de Programação de Tripulações, Transporte Público, *Adaptive Large Neighborhood Search*.

Área principal: Logística e Transportes, Metaheurísticas

ABSTRACT

The crew scheduling problem (PPT) consists of minimizing the total cost involved with the crews that will drive a fleet of an urban bus public transportation company. This work presents a PPT resolution through the Adaptive Large Neighborhood Search (ALNS) heuristic, which is characterized by the destruction and repair of the solution, through removal and insertion heuristics. The pair of heuristics choose in each iteration is based on dynamically adjusted weights, computed bearing in mind the efficiency of each method over the search. The implementation was tested with the data of a company that operates in Belo Horizonte-MG, and has generated solutions better than those adopted by the company and also better than the solutions generated by other methods on the literature.

KEYWORDS. Crew Scheduling Problem, Public Transport, Adaptive Large Neighborhood Search.

Main area: Logistic & Transport, Metaheuristics

1. Introdução

O planejamento do Sistema de Transporte Público Urbano é um processo complexo e dividido em etapas, que envolve tanto as empresas de transporte urbano quanto o poder público. Este planejamento é composto por cinco etapas: *i*) a definição das rotas dos ônibus, *ii*) a criação dos quadros de horários das linhas para todos os dias da semana, *iii*) a programação dos veículos que realizarão as viagens, *iv*) a programação diária das tripulações que conduzirão a frota em operação, e *v*) o rodízio das tripulações ao longo de um dado período de planejamento (Bodin *et al.*, 1983).

Dentre as etapas do planejamento, as empresas devem realizar a programação dos veículos, das tripulações e o rodízio das tripulações. Ou seja, definir o tamanho da frota, o número de tripulações (motorista e cobrador) e os períodos de trabalho e de folga das tripulações de forma que as viagens sejam executadas com o menor custo possível. Esta é uma tarefa de grande complexidade para a qual podem ser utilizados métodos de otimização na sua resolução (Wren, 2004).

A realização das viagens leva à definição de uma escala diária de trabalho para os motoristas e cobradores, denominada Programação de Tripulações. Dessa forma, o *Problema de Programação de Tripulações* (PPT) consiste em determinar o número mínimo de tripulações e atribuir a elas as viagens que devem ser executadas em um determinado dia de trabalho, sem que haja sobreposição entre as viagens. O objetivo é minimizar o número de tripulações e o total de horas extras contidas na escala diária (Bodin *et al.*, 1983).

O PPT tem como dados de entrada a programação dos veículos, definida para os dias úteis, os sábados e os domingos/feriados. A programação dos veículos define os *blocos dos veículos*, ou seja, o conjunto das viagens executadas por cada veículo da frota. Os blocos são divididos em *tarefas*, que são um conjunto de viagens que devem ser executadas por uma tripulação, visto que não há condições de troca da tripulação. As condições necessárias para a troca de duas tripulações definem uma *oportunidade de troca* (OT), ou seja, um intervalo de tempo mínimo entre duas viagens, que deve ocorrer em um local apropriado (Silva *et al.*, 2006). As tarefas apresentam um ponto inicial e final, e um horário de início e de término. Além disso, as tarefas estão associadas a um veículo. Um conjunto de tarefas executadas por uma tripulação constitui uma *jornada*. Desta forma, uma tripulação está associada a uma jornada e vice-versa e, portanto, são considerados sinônimos.

As jornadas podem ser classificadas em *simples* ou *dupla pegada*. Uma jornada do tipo dupla pegada é dividida em duas etapas, com um intervalo superior a duas horas entre elas. Isso ocorre devido aos picos de demanda de viagens, comuns nos dias úteis. O intervalo superior a duas horas não é considerado como hora trabalhada. Uma jornada simples não apresenta uma interrupção deste tipo. Independentemente do tipo de jornada, as horas extras correspondem ao tempo trabalhado que excede o tempo de duração normal de uma jornada diária.

A construção das jornadas deve satisfazer um conjunto de restrições legais e operacionais, o que torna o PPT um problema *NP-Completo* (Fischetti *et al*., 1987). Na resolução do PPT, para o caso estudado, foram consideradas as seguintes restrições: 1) as jornadas tem uma remuneração fixa por seis horas e quarenta minutos de trabalho, que define a duração normal de trabalho; 2) as jornadas não podem ter mais que duas horas extras por dia; 3) o intervalo entre o fim de uma jornada e o seu início no dia seguinte deve ser de, pelo menos, onze horas; 4) o número de troca de veículos, realizadas em uma jornada, é limitado; e 5) a troca de terminal só é permitida entre as duas pegadas de uma jornada do tipo dupla pegada.

Neste trabalho é apresentada a resolução do PPT por meio da heurística *Adaptive Large Neighborhood Search* (ALNS). A ALNS é caracterizada pela destruição e reparo da solução corrente, através de várias heurísticas de remoção e inserção de tarefas. A escolha das heurísticas utilizadas em cada iteração é realizada através de um peso ajustado dinamicamente que se baseia na eficiência de cada heurística ao longo da busca. A ALNS foi testada com dados reais de uma empresa de médio porte que atua na cidade de Belo Horizonte, MG, e as soluções alcançadas

superaram as soluções adotadas na prática pela empresa e também as soluções obtidas por outros métodos da literatura.

Este trabalho está dividido da seguinte forma: na Seção 2 é apresentada uma breve revisão bibliográfica sobre o PPT. Na Seção 3 é descrito como a ALNS foi implementada para resolver o PPT. Na Seção 4 são apresentados os resultados obtidos nos experimentos computacionais e a discussão destes resultados. Finalmente, na Seção 5, são apresentadas as conclusões do trabalho.

2. Revisão Bibliográfica

A abordagem exata clássica formula o PPT como um problema de cobertura ou de particionamento (*set covering* ou *set partitioning model*) e utiliza a técnica de geração de colunas para resolvê-lo. Lourenço *et al.* (2001) formulam o PPT como um problema de cobertura, considerando várias funções objetivo. Os autores apresentam metaheurísticas multi-objetivo para resolver o PPT visto que na prática, o problema apresenta diferentes objetivos conflitantes. As metaheurísticas se baseiam na Busca Tabu e nos Algoritmos Genéticos (AG), e tem como procedimento interno o GRASP mono-objetivo. Para todos os problemas considerados, a Busca Tabu multi-objetivo e os AG multi-objetivos geraram bons resultados em um tempo razoável.

Silva e Cunha (2010) apresentaram uma nova abordagem para o PPT baseada na metaheurística GRASP, tendo como busca local o *Very Large-Scale Neighborhood Search* (VLNS) (Ahuja *et al.,* 2002). Nesta abordagem, uma solução vizinha é obtida pela realocação e/ou troca das tarefas entre duas ou mais jornadas, podendo envolver todas as jornadas na troca. Para pesquisar a vizinhança, é construído um *grafo de melhoria*. A partir deste grafo é feita uma busca por um *ciclo válido*, que é um ciclo negativo que leva à redução do custo da solução. Ao encontrar um ciclo válido, são realizadas as trocas contidas no ciclo e o grafo é atualizado. A busca se encerra quando nenhum ciclo negativo for encontrado. Os testes computacionais mostraram que a abordagem produz melhoras significativas em comparação com a solução adotada pela empresa.

Reis e Silva (2012) exploram diferentes métodos de Busca Local associados à metaheurística *Variable Neighborhood Search* (VNS) para resolver o PPT. As soluções vizinhas da solução corrente são obtidas através da realocação e/ou trocas de tarefas entre duas tripulações e os métodos de Busca Local adotados foram o *Variable Neighborhood Descent* (VND) e o VLNS (Silva e Cunha, 2010). Foram considerados cenários que permitem uma ou duas trocas de veículos por jornada e ainda cenários onde as duplas pegadas são mais caras, para controlar a sua ocorrência. Ambas as técnicas superaram a solução da empresa e a técnica VNS-VLNS superou o VNS clássico.

Chen e Shen (2013) apresentam uma nova estratégia de geração de colunas para tratar o PPT. Tradicionalmente, as colunas a serem adicionadas ao problema principal são determinadas pela resolução de subproblemas com alto custo computacional. Por isso, os autores propõem que um conjunto razoavelmente grande de jornadas seja considerado. Estas jornadas, chamado *shift-pool*, são pré-compiladas com características desejadas. Durante o processo de geração de colunas, o subproblema é resolvido até que não existam mais colunas com reduções de custo na *shift-pool*. Esta ideia permite reduzir o tempo de processamento para resolver o PPT. Nos experimentos são consideradas instâncias de até 701 viagens, 55 blocos de veículos e duas garagens. Os resultados mostram que o algoritmo proposto supera a geração de colunas tradicional.

Souza (2014) apresenta um modelo híbrido de Programação Linear Inteira em conjunto com a metaheurística *Late Acceptance Hill Climbing* (LAHC) para resolver o PPT. É apresentado um modelo compacto que, ao contrário dos modelos clássicos de particionamento ou de cobertura, onde jornadas devem ser geradas e informadas a *priori*, o modelo é composto por restrições que permite a resolução do PPT a partir das tarefas a serem realizadas. A LAHC é utilizada com o objetivo de resolver problemas de maiores dimensões. Assim, partindo de uma solução gerada pela LAHC, os componentes da solução são particionados em conjuntos menores que são submetidos ao método

exato. O modelo foi testado com problemas reais e se mostraram eficientes na geração das soluções, mas em um tempo elevado de processamento.

Silva e Silva (2015) aplicam a metaheurística *Guided Local Search* (GLS) para resolver o PPT. A GLS é caracterizada pela construção de uma sequência de soluções, realizando uma busca local na solução corrente, e se diferencia das demais metaheurísticas por penalizar as características indesejáveis na solução corrente como forma de escapar de ótimos locais. Desta forma, ao encontrar um ótimo local, a GLS seleciona alguns atributos da solução e os penalizam, fazendo que a solução deixe de ser um ótimo local e que a busca possa se mover para outro ótimo local. O processo se repete até que algum critério de parada seja satisfeito. A GLS foi combinada com o método de busca local *Variable Neighborhood Descent* (VND), produzindo resultados similares aos da literatura.

Song *et al.* (2015) tratam o PPT com um AG aperfeiçoado com recombinação de genes. A recombinação de genes é responsável pela reconstrução de jornadas, capaz de reduzir o número de jornadas na solução. São utilizadas listas tabu e não tabu para a construção das soluções iniciais. O método da roleta foi adotado para selecionar os indivíduos para o *crossover*, caracterizado por dois pontos de corte. A mutação consiste na reconstrução de *n* jornadas escolhidas aleatoriamente. Elas são divididas em segmentos que são reconstruídos e substituídos. O método foi capaz de reduzir mais de 30 jornadas quando tem-se um grande número de tarefas, sendo um algoritmo muito rápido.

A heurística *Adaptive Large Neighborhood Search* (ALNS) foi proposta por Ropke e Pisinger (2006a) para resolver o Problema de Coleta e Entrega com Janelas de Tempo. Ela parte de uma solução inicial viável e aplica mecanismo do tipo destrói-reconstrói para encontrar soluções melhores. As heurísticas de remoção e inserção são escolhidas a partir de estatísticas recolhidas durante o processo. São usadas três heurísticas de remoção: *Shaw*, *Random* e *Worst* e duas heurísticas de inserção: *Greedy* e *Regret*. Em cada iteração, uma heurística de remoção e uma de inserção é escolhida independentemente e o par é aplicado. Todas heurísticas são utilizadas na busca e a seleção do par é feita levando em consideração os pesos individuais de cada heurística. Os pesos são ajustados dinamicamente dado o sucesso da heurística. A ALNS foi capaz de melhorar as melhores soluções conhecidas na literatura para mais de 50% dos problemas, indicando a vantagem de utilizar várias heurísticas de remoção e inserção concorrentemente ao invés de apenas um par de heurísticas.

Os mesmos autores integraram, posteriormente, novas heurísticas de remoção à ALNS. Três novas heurísticas de remoção: *Cluster*, *Neighbor Graph* e *Request Graph* foram propostas e o problema passou a cobrir uma classe de Problemas de Roteamento de Veículos (Ropke e Pisinger, 2006b). Posteriormente, abrangendo diferentes variantes do Problema de Roteamento de Veículos, os autores propuseram três novas heurísticas de remoção: *Time-oriented*, *Node-pair removal* e *Historical request-pair* (Pisinger e Ropke, 2007).

Como o problema de coleta e entrega com janela de tempo é semelhante ao PPT, levanta-se a hipótese se a ALNS pode ser eficiente na sua resolução. Desta forma, este trabalho apresenta as adaptações realizadas na ALNS para resolver o PPT, assim como os resultados obtidos

3. *Adaptive Large Neighborhood Search* Aplicada ao PPT

A ALNS inicia sua busca a partir de uma solução inicial viável e opera com diferentes métodos que destroem e reparam a solução corrente. No ALNS várias heurísticas de remoção e inserção são utilizadas na mesma busca. Um peso é inicialmente atribuído a cada heurística com o intuito de se encontrar o par mais eficiente para o problema. Os pesos são ajustados dinamicamente, baseados na eficiência de cada método ao longo da busca.

3.1. Representação da Solução

Para o PPT, uma solução é representada por um conjunto de jornadas, e cada jornada contém a sequência das tarefas a serem realizadas pela tripulação responsável. Esta representação possibilita aplicar os mecanismos de remoção e reinserção inerentes à ALNS.

3.2. Solução Inicial

A ALNS inicia sua busca a partir de uma solução viável. A solução inicial do PPT foi gerada de maneira gulosa. Inicialmente as tarefas do conjunto de tarefas são ordenadas pelo horário de início crescente. A partir da primeira tarefa t_i do conjunto, inicia-se a alocação das tarefas. A tarefa é incluída na primeira jornada. A próxima tarefa t_j, a ser incluída, é aquela que leva ao menor acréscimo na função objetivo e respeita as seguintes restrições: *i*) pertence ao mesmo veículo de operação de t_i; *ii*) o horário de início de t_j é maior ou igual ao horário de término de t_i; e *iii*) o ponto de início de t_j é igual ao ponto de término de t_i. A última tarefa alocada passa a ser a nova tarefa t_i, e o processo se repete enquanto o tempo máximo para uma jornada não for atingido. Caso o tempo máximo permitido para uma jornada exceda o tempo normal, esta alocação resulta em horas extras para a jornada. E caso o tempo máximo permitido para uma jornada seja atingido, uma nova jornada é inicializada e o processo se repete até que todas as tarefas tenham sido alocadas.

3.3. Função Objetivo

Considere $j_i, i = 1, \ldots, n$, as jornadas de uma solução *s*, a serem realizadas durante um determinado dia. Então, o custo diário de execução das jornadas é calculado pela expressão:

$$Custo\ (s) = \sum_{i=1}^{n} CustoFixo \quad + w_i \times HE\ (j_i) + w_2 \times DP\ (j_i) \tag{1}$$

O *CustoFixo* está associado ao custo de uma dupla de funcionários, $HE(j_i)$ é o tempo total de horas extras da jornada j_i e $DP(j_i) = 1$ se j_i for uma jornada do tipo dupla pegada, caso contrário $DP(j_i) = 0$. As constantes w_1 e w_2 são ajustadas de acordo com a realidade da empresa.

O custo de uma solução do PPT, dado pela equação (1) foi avaliado considerando os seguintes valores para os parâmetros: *CustoFixo* = 10.000, w_1 = 4 e, w_2 = 600 e w_2 = 5.000, de modo que o método desenvolvido possa ser comparado com outros autores que utilizaram a mesma base de dados e ponderação dos parâmetros da Função Objetivo (FO). Dessa forma, o custo da tripulação tem um valor elevado para que a redução no número de duplas seja o principal objetivo, seguindo-se pela redução das duplas pegadas e finalmente, das horas extras. O valor da FO para a solução da empresa foi calculado pela expresão (1), definida para o ALNS.

3.4. Heurísticas Destrutivas

Foram implementadas quatro heurísticas destrutivas, sendo três delas baseadas naquelas propostas por Ropke e Pisinger (2006a), e uma delas foi proposta neste trabalho. No PPT, os elementos removidos e reinseridos na solução são as tarefas das tripulações. Assim, uma heurística destrutiva deve remover uma quantidade pré-definida *q* de tarefas de uma dada solução.

3.4.1. Remoção Randômica

A Remoção Randômica é caracterizada pela remoção de *q* tarefas escolhidas aleatoriamente. Inicialmente é sorteada uma jornada e dela é removida uma tarefa, também escolhida aleatoriamente.

3.4.2. Remoção de Pior Posição

Na Remoção de Pior Posição, a cada iteração é removida a tarefa que produz a maior diferença entre o valor da FO com a tarefa e o valor da FO sem a tarefa na solução. O *pior custo* de uma tarefa *i* em uma solução *s* é dado por $custo\_pior(i, s) = f(s) - f_{-i}(s)$, onde $f_{-i}(s)$ é o custo da solução *s* sem a tarefa *i*. Esta remoção tenta realocar tarefas que causam grande impacto na FO.

3.4.3. Remoção Shaw

A Remoção *Shaw* tem como objetivo remover tarefas similares, permitindo que sejam facilmente realocadas, na esperança de gerar soluções melhores. A semelhança de duas tarefas é computada pela medida de *similaridade*. Duas tarefas são similares se elas iniciarem e finalizarem no mesmo local, pois é impossível realizar a troca de tarefas que com locais distintos de início e término. A similaridade entre duas tarefas *i* e *j* para o PPT é dada por:

$$similaridade(i,j) = \alpha\left(|st_i - st_j| + |et_i - et_j|\right) + \beta\left(|(et_i - st_i) - (et_j - st_j)|\right) \quad (2)$$

Em (2), st_i corresponde à hora inicial e et_i à hora final da tarefa i. A primeira parcela de (2) considera a proximidade do horário do início e do final das tarefas, enquanto a segunda parcela considera a proximidade da duração das duas tarefas. Desta forma, quanto menor o valor de (2), maior a similaridade entre as tarefas i e j. Os pesos α e β são usados para calibrar a similaridade.

O procedimento remove aleatoriamente a primeira tarefa. As demais tarefas são escolhidas de acordo com a similaridade em relação a uma tarefa já removida. Determinadas tarefas podem não ter q-1 tarefas similares, por não existirem q-1 tarefas que iniciam e terminam no mesmo local. Neste caso, a quantidade q de tarefas removida passa a ser a quantidade possível de remoção.

3.4.4. Remoção Média

Como um dos objetivos do PPT é minimizar o número de horas extras, é interessante que as jornadas tenham a durações semelhantes. Assim, é proposto neste trabalho o método de Remoção Média. Nesta remoção, são removidas tarefas das jornadas cujos tempos de duração excedem a duração média. São permitidas remoções da primeira tarefa ou da última tarefa da jornada. A escolha é aleatória a cada remoção. Não é removida uma tarefa no interior da jornada, visto que ela não diminui a duração da jornada e pode aumentar sua ociosidade interna.

3.5. Heurísticas Construtivas

Para reparar as soluções parcialmente destruídas, duas heurísticas construtivas foram empregadas. As heurísticas foram adaptadas daquelas propostas por Ropke e Pisinger (2006a) e Potvin e Rousseau (1993). No contexto do PPT, *inserir uma tarefa em uma posição* é o mesmo que inserir uma tarefa em uma jornada, visto que não é possível alocá-la em mais de uma posição na jornada, devido à ordem cronológica das tarefas. Quando a inserção de novas tarefas nas jornadas existentes não for possível, é criada uma nova jornada a partir da tarefa a ser inserida.

Durante a reconstrução das jornadas, permite-se que sejam realizadas trocas de veículos. Isso possibilita ampliar o espaço de busca. Uma quantidade máxima de trocas de veículos em uma jornada é definida a *priori*. Para o caso estudado, permite-se até 2 trocas de veículo por jornada.

3.5.1. Inserção Gulosa

O método de Inserção Gulosa insere uma tarefa em sua posição mais barata na solução. Para cada tarefa i, é verificada, dentre as possibilidades de inserção, aquela que resulta no menor custo. Esta posição é denominada *posição de custo mínimo*, c_i. Por fim, é escolhida a tarefa i com o menor valor de c_i e é realizada a inserção em sua posição de custo mínimo.

3.5.2. Inserção de Arrependimento

As heurísticas de arrependimento fazem uma espécie de avaliação do tipo "olhar à frente" ao selecionar uma tarefa para inserção. Considere $f_\Delta(i,k)$ como a mudança no valor da FO causada pela inserção da tarefa i em sua *k-ésima* posição válida mais barata. Então, o valor da inserção 2-arrependimento é dado por $c_i^* = f_\Delta(i,2) - f_\Delta(i,1)$. Em cada iteração, é inserida a tarefa i que maximize c_i^*. Desta forma, é escolhida a tarefa que produzir o maior impacto na FO caso ela não seja alocada na melhor posição e tenha que ser inserida na segunda melhor posição.

3.6. Escolha das Heurísticas de Remoção e Inserção

Definidas as heurísticas que destroem e reconstroem uma solução, o próximo passo consiste em escolher um par de heurísticas para modificar a solução corrente. Um único par de heurísticas poderia ser utilizado durante toda a busca. No entanto, é interessante que todas as heurísticas sejam utilizadas, visto que cada uma delas pode ser mais adequada em um determinado momento da busca. Acredita-se que esta alternância resulta numa aplicação mais robusta (Ropke e Pisinger, 2006a).

A escolha das heurísticas que formam o par é independente, e a seleção de cada uma, durante a busca, é feita pelo princípio de seleção da roleta. A cada heurística é atribuído um peso, e o sorteio

é dado em função de sua probabilidade de escolha. Considerando um conjunto de k heurísticas, com pesos w_i, $i \in \{1, 2, \ldots, k\}$, uma heurística j é selecionada com probabilidade:

$$w_j \Big/ \sum_{i=1}^{k} w_i \tag{3}$$

Suponha um gráfico de pizza onde a cada heurística é atribuído um espaço proporcional ao seu peso. Uma roleta exterior é colocada sobre o gráfico e uma heurística é selecionada. Quanto maior o peso da heurística, maior seu espaço no gráfico e, portanto, maior sua chance de ser escolhida. O peso está diretamente relacionado com a eficiência da heurística em melhorar a solução.

3.7. Ajuste Adaptativo dos Pesos

Os *pesos* das heurísticas representam a eficiência de cada uma durante o processo de busca da ALNS. Os pesos são ajustados usando estatísticas das iterações anteriores. Uma *pontuação* é mantida para cada heurística, que mede seu desempenho em um determinado número de iterações. Quanto maior a pontuação da heurística, maior sua eficiência. A busca é dividida em *segmentos*, que por sua vez, são divididos em iterações ou tempo de processamento. Assim, definem-se os critérios de parada do algoritmo. No início de cada segmento, a pontuação de cada heurística é definida como zero, e é aumentada durante a busca em três situações diferentes. No primeiro caso, aumenta-se σ_1 quando a heurística é capaz de encontrar uma nova solução global *s**. Aumenta-se em σ_2 quando a heurística é capaz de encontrar uma solução *s'* ainda não encontrada e que seja melhor do que a solução atual *s*. No último caso, aumenta-se em σ_3 quando a heurística é capaz de encontrar uma solução *s'* ainda não encontrada e que seja pior que a solução atual *s*, porém ainda aceita pelo critério de aceitação. Como a situação associada a σ_1 é mais atraente que a situação caracterizada por σ_2 que, por sua vez, é mais atraente que a situação caracterizada por σ_3, é razoável que $\sigma_1 > \sigma_2 > \sigma_3$. Apesar de serem preferíveis as heurísticas que melhoram a solução, é também interessante utilizar heurísticas que possam diversificar a busca, como no último caso, o que encoraja as heurísticas a explorarem novas regiões do espaço de busca.

Em cada iteração da ALNS são aplicadas uma heurística de remoção e uma heurística de inserção. Como ambas são responsáveis pela geração de uma nova solução, a pontuação das mesmas é atualizada na mesma quantidade, pois não é possível garantir qual das duas foi o fator de sucesso da operação. No fim de cada segmento, os pesos das heurísticas são atualizados levando em consideração a pontuação obtida no segmento corrente.

Assim como as pontuações são inicializadas com o valor zero para cada heurística, os respectivos pesos são inicializados com valor 1. No final de cada segmento, novos pesos são calculados utilizando as pontuações salvas. As pontuações, por sua vez, são atualizadas no fim de cada iteração, e são zeradas ao iniciar um novo segmento. Sendo $w_{i,j}$ o peso da heurística *i* utilizado no segmento *j*, o peso da heurística *i* a ser utilizado no segmento *j*+1 é calculado como:

$$w_{i,j+1} = w_{i,j}(1 - r) + r\frac{\pi_i}{\theta_i} \tag{4}$$

Em (4), π_i é a pontuação obtida pela heurística *i* durante seu último segmento, e θ_i é a quantidade de vezes que a heurística *i* foi utilizada durante o último segmento. O *fator de reação r* controla a influência dos pesos anteriores nos próximos pesos. Se $r = 0$, os pesos anteriores são repetidos. Se $r = 1$, os pesos do segmento anterior são totalmente descartados. Um valor entre 0 e 1 para *r* resulta em novos pesos que consideram o desempenho das heurísticas no segmento anterior.

3.8. Critério de Aceitação

Foi utilizado o critério de aceitação da metaheurística *Simulated Annealing*. Dessa forma, uma solução *s'* será aceita se for melhor do que *s*. Se *s'* for pior do que *s*, *s'* será aceita com probabilidade $e^{-(f(s')-f(s))/T}$, onde $T > 0$ representa a temperatura atual. A temperatura *T* inicia em t_0 e diminui a cada iteração pela expressão $T = T \times c$, onde $0 < c < 1$ é a *taxa de resfriamento*. Como

uma boa escolha de t_0 deve estar relacionada ao problema, foi utilizado um método para a obtenção da temperatura inicial que leva em consideração as características de cada instância. Neste trabalho a temperatura inicial parte de um valor baixo e é aumentada a uma taxa de 10% até que pelo menos 70% das soluções vizinhas fossem aceitas.

3.9. Método para Minimizar o Número de Jornadas

O método utilizado para minimizar o número de jornadas no PPT parte de uma solução inicial viável s_i, com um determinado número de jornadas. Para cada jornada, todas as suas tarefas são removidas e a heurística de Inserção Gulosa tenta reinseri-las nas jornadas existentes, criando uma nova solução *s'*. Se, ao final da reinserção, o número de jornadas de *s'* for menor do que em s_i, *s'* é aceita e a busca se reinicia a partir da primeira jornada da nova solução. Caso contrário, passa-se para a jornada seguinte. O procedimento é encerrado quando todas as jornadas foram consideradas no processo de destruição e reconstrução da solução, e o número de jornadas não foi reduzido.

O método de minimização de jornadas é aplicado na solução inicial, antes de realizar a busca ALNS. Assim, espera-se que a ALNS possa reduzir os custos variáveis da solução, visto que o número de jornadas já foi minimizado.

4. Experimentos Computacionais

Para os experimentos computacionais, foram utilizados dados de uma empresa de transporte público que opera em Belo Horizonte, MG. Os testes foram realizados em um computador com Intel Core i7-4770, 8 GB de RAM, Ubuntu 16.04, e o algoritmo foi desenvolvido em C++.

Os dados fornecidos pela empresa são: a quantidade de tarefas a ser executada em cada dia útil, no sábado e no domingo de uma semana, com os seus horários de início e término, ponto inicial e final, e veículo de operação. Na Tabela 1 consta, para cada tipo de dia, a quantidade de tarefas a serem executadas em cada dia da semana.

Tabela 1. Quantidade de tarefas a serem executadas nos dias úteis, sábados e domingos

	Seg	**Ter**	**Qua**	**Qui**	**Sex**	**Sab**	**Dom**
Total de Tarefas	705	674	814	872	787	644	345

4.1. Calibragem dos Parâmetros

A heurística ALNS conta com uma série de parâmetros a serem definidos. Primeiro, deve-se definir a quantidade q de tarefas a serem removidas e reinseridas na solução, a qual é controlada pelo parâmetro ξ. Em cada iteração, escolhe-se um número aleatório q que satisfaz $0{,}008m \leq q \leq \xi m$, onde m representa a quantidade de tarefas do problema. O critério de aceitação de uma solução é controlado por uma temperatura inicial t_0, resfriada a uma taxa c. A heurística Remoção *Shaw* é controlada pelos parâmetros α, β e p_{shaw}, enquanto a heurística Remoção de Pior Posição é controlada apenas pelo parâmetro p_{pior}. Também devem ser definidos os parâmetros σ_1, σ_2, σ_3 e r, utilizados no ajuste adaptativo dos pesos. O critério de parada da ALNS é definido pela quantidade de segmentos, cada um constituído por 100 iterações.

Dentre os parâmetros considerados, calibrou-se os parâmetros t_0, ξ e r, enquanto o restante dos parâmetros citados foram definidos de acordo com Ropke e Pisinger (2006a). Após a realização de diversos testes computacionais e do teste estatístico de Kruskal-Wallis, chegou-se à configuração final dos parâmetros, representada pelo vetor (%t_0, ξ, c, α, β, p_{shaw}, p_{pior}, σ_1, σ_2, σ_3, r) = (0.781, 0.025, 0.99975, 1, 1, 6, 6, 20, 10, 5, 0.8) para os dias úteis e por (%t_0, ξ, c, α, β, p_{shaw}, p_{pior}, σ_1, σ_2, σ_3, r) = (1.563, 0.05, 0.99975, 1, 1, 6, 6, 20, 10, 5, 0.8) para o fim de semana. Essas foram as configurações utilizadas para medir o desempenho da ALNS implementada.

4.2. Resultados

Quatro cenários distintos foram analisados: os cenários $1TV_{600}$ e $1TV_{5.000}$ que permitem, no máximo, uma troca de veículos por jornada e as duplas pegadas possuem peso 600 e 5.000, respectivamente, e os cenários $2TV_{600}$ e $2TV_{5.000}$ que permitem até duas trocas de veículos por jornadas com os mesmos pesos das duplas pegadas. Para cada problema e cenário, a ALNS foi executada 10 vezes, cada uma limitada a 1 hora de duração. Os resultados obtidos foram comparados com aqueles obtidos pelos métodos GRASP (Silva e Cunha, 2010), VNS-VLNS (Reis e Silva, 2012) e GLS (Silva e Silva, 2015). As diferentes abordagens consideradas em cada cenário de testes são justificadas pelas configurações de cenário adotadas pelos autores dos métodos considerados.

Nas Tabelas 2 e 3 são apresentados os resultados obtidos para os cenários de peso 600 e 5.000 para as jornadas de dupla pegada, respectivamente. Para a ALNS são apresentados o valor da FO da melhor solução (Melhor FO) obtida, e seu detalhamento em quantidade de jornadas (#Jornadas), de duplas pegadas (#DP) e horas extras (HE). Na última linha é apresentado o *gap,* dado por $gap = (s_{alns} - s_{melhor})/s_{melhor}$, onde s_{alns} é o valor da FO da melhor solução obtida pela ALNS e s_{melhor} é o valor da FO da melhor solução conhecida para o problema e cenário de testes. Um *gap* negativo (em verde) significa que a ALNS superou a melhor solução, em determinada porcentagem. Um *gap* positivo (em vermelho) significa que a ALNS gerou solução pior que a melhor solução, em determinada porcentagem. Os valores em negrito representam as melhores soluções encontradas por problema.

Tabela 2. Resultados obtidos pela ALNS, GRASP, VNS-VLNSe GLS no cenário $1TV_{600}$

	Dados	Segunda	Terça	Quarta	Quinta	Sexta	Sábado	Domingo
ALNS	**Melhor FO**	**1.204.500**	**1.162.832**	1.385.804	**1.505.896**	**1.433.608**	**1.050.576**	565.812
	#Jornadas	117	113	135	147	141	101	55
	#DP	21	16	18	16	14	28	10
	HE	91:15:00	96:48:00	104:11:00	109:34:00	105:02:00	99:04:00	40:53:00
GRASP	**Melhor FO**	1.264.836	1.265.880	1.462.384	1.564.420	1.510.520	1.131.272	590.716
	#Jornadas	124	125	144	154	149	111	58
	#DP	14	7	11	12	10	15	7
	HE	68:29:00	48:40:00	65:46:00	71:45:00	60:30:00	51:08:00	27:09:00
VNS-VLNS	**Melhor FO**	1.223.556	1.191.952	**1.380.000**	1.517.168	1.435.332	1.089.764	**562.444**
	#Jornadas	120	117	135	149	141	107	55
	#DP	19	14	19	17	14	10	7
	HE	50:39:00	56:28:00	77:30:00	70:42:00	70:33:00	57:21:00	34:21:00
GLS	**Melhor FO**	1.253.532	1.222.920	1.417.292	1.557.100	1.474.588	1.104.096	581.468
	#Jornadas	123	120	139	153	145	108	57
	#DP	23	23	14	22	19	17	9
	HE	40:33:00	38:00:00	78:43:00	57:55:00	54:57:00	57:54:00	74:14:00
Empresa	**Melhor FO**	1.364.404	1.322.420	1.518.460	1.651.256	1.576.564	1.253.100	686.468
	#Jornadas	134	130	149	162	155	124	68
	#DP	6	3	5	4	1	0	0
	HE	86:41:00	85:55:00	106:05:00	120:14:00	108:11:00	54:35:00	26:57:00
GAP		**-1,58%**	**-2,50%**	**0,42%**	**-0,75%**	**-0,12%**	**-3,73%**	**0,60%**

Ao analisar o primeiro cenário, a ALNS gerou as melhores soluções para os problemas da segunda-feira, terça-feira, quinta-feira, sexta-feira e sábado. Para a quarta-feira e domingo, as soluções obtidas pela ALNS foram, no máximo, 0,6% piores que as melhores soluções. Também nota-se que a maioria das soluções geradas pela ALNS apresentam um número menor de jornadas do que dos demais métodos. Essa redução se explica pelo aumento de horas extras, que representa o *trade-off* do problema: ao diminuir o número de jornadas, há um aumento de horas extras, uma vez que um número menor de tripulações deve realizar as mesmas tarefas.

Tabela 3. Resultados obtidos pela ALNS, VNS-VLNS e GLS no cenário $1TV_{5.000}$

	Dados	Segunda	Terça	Quarta	Quinta	Sexta	Sábado	Domingo
ALNS	**Melhor FO**	**1.256.628**	**1.202.692**	**1.440.876**	**1.565.016**	1.471.920	**1.127.876**	602.836
	#Jornadas	120	115	138	151	141	105	57
	#DP	7	6	7	6	7	11	5
	HE	90:07	94:33	107:49	104:14	112:10	95:19	32:39
VNS-VLNS	**Melhor FO**	1.270.628	1.213.176	1.471.176	1.583.644	**1.471.100**	1.152.552	**601.296**
	#Jornadas	120	114	140	148	139	109	57
	#DP	11	11	11	17	12	10	5
	HE	65:07	75:44	67:24	77:41	87:55	52:18	27:35
GLS	**Melhor FO**	1.276.280	1.216.340	1.455.988	1.581.920	1.478.832	1.146.440	601.560
	#Jornadas	120	116	139	150	142	110	57
	#DP	12	8	10	13	8	7	5
	HE	67:50	68:05	66:37	70:30	78:28	47:40	27:20
Empresa	**Melhor FO**	1.390.804	1.335.620	1.540.460	1.668.856	1.580.964	1.253.100	686.468
	#Jornadas	134	130	149	162	155	124	68
	#DP	6	3	5	4	1	0	0
	HE	86:41	85:55	106:05	120:14	108:11	54:35	26:57
GAP		**-1,102%**	**-0,864%**	**-1,038%**	**-1,069%**	**+0,056%**	**-1,619%**	**+0,256%**

Considerando o segundo cenário, há uma redução no número de jornadas com dupla pegada nas soluções, se comparadas às soluções do cenário anterior. Esta estratégia fez com que as heurísticas gerassem soluções com o número de duplas pegadas próximas ao valor desejado. Analisando a Tabela 3, a ALNS gerou os melhores resultados para os problemas da segunda-feira, terça-feira, quarta-feira, quinta-feira e sábado. Para a sexta-feira e domingo, as soluções do ALNS foram, no máximo, 0,3% piores que as melhores soluções.

Na prática, é aceitável que as tripulações realizem, no máximo, duas trocas de veículos por jornada. Essas trocas são limitadas devido à dificuldade de controlar danos materais nos veículos, causados pelas tripulações. Os resultados deste cenário são apresentados nas Tabelas 4 e 5.

Tabela 4. Resultados obtidos pela ALNS, GRASP e VNS-VLNS no cenário $2TV_{600}$

	Dados	Segunda	Terça	Quarta	Quinta	Sexta	Sábado	Domingo
ALNS	**Melhor FO**	**1.210.492**	**1.152.376**	**1.389.968**	**1.510.252**	**1.425.480**	**1.055.260**	565.608
	#Jornadas	118	112	136	148	139	102	55
	#DP	17	17	13	13	15	20	10
	HE	84:33	92:24	92:22	93:33	110:20	96:55	40:02
GRASP	**Melhor FO**	1.264.386	1.265.916	1.457.172	1.563.532	1.508.884	1.131.828	591.060
	#Jornadas	124	125	143	154	149	111	58
	#DP	14	7	12	11	9	14	8
	HE	66:32	48:49	83:13	70:33	56:11	55:57	26:05
VNS-VLNS	**Melhor FO**	1.223.556	1.189.408	1.393.920	1.515.196	1.437.060	1.089.764	**562.444**
	#Jornadas	120	117	137	149	141	107	55
	#DP	19	10	15	17	17	10	7
	HE	50:39	55:52	62:10	53:36	56:18	43:12	34:21
Empresa	**Melhor FO**	1.364.404	1.322.420	1.518.460	1.651.256	1.576.564	1.253.100	686.468
	#Jornadas	134	130	149	162	155	124	68
	#DP	6	3	5	4	1	0	0
	HE	86:41	85:55	106:05	120:14	108:11	54:35	26:57
GAP		**-1,079%**	**-3,214%**	**-0,284%**	**-0,327%**	**-0,812%**	**-3,270%**	**+0,563%**

Ao analisar a Tabela 4, a ALNS gerou as melhores soluções para todos os dias úteis e para o sábado neste cenário. Quanto ao domingo, a solução do ALNS foi apenas 0,6% pior que a melhor solução. Assim como no cenário $1TV_{600}$, a redução no número de jornadas da solução gerada pela ALNS, em relação aos demais métodos, é justificada pelo aumento das horas extras em cada solução.

Tabela 5. Resultados obtidos pela ALNS e VNS-VLNS no cenário $2TV_{5.000}$

	Dados	Segunda	Terça	Quarta	Quinta	Sexta	Sábado	Domingo
ALNS	**Melhor FO**	**1.252.288**	**1.200.932**	**1.435.604**	**1.573.848**	1.471.984	**1.108.708**	599.496
	#Jornadas	121	116	138	152	141	104	56
	#DP	4	4	6	6	7	9	6
	HE	92:52	87:13	106:41	99:22	112:26	98:47	39:34
VNS-VLNS	**Melhor FO**	1.264.724	1.220.112	1.449.952	1.580.160	**1.461.288**	1.141.916	**597.496**
	#Jornadas	121	118	139	150	139	110	58
	#DP	12	8	11	14	10	11	6
	HE	51:34	58:25	58:19	58:33	66:05	49:39	27:12
Empresa	**Melhor FO**	1.390.804	1.335.620	1.540.460	1.668.856	1.580.964	1.253.100	686.468
	#Jornadas	134	130	149	162	155	124	68
	#DP	6	3	5	4	1	0	0
	HE	86:41	85:55	106:05	120:14	108:11	54:35	26:57
GAP		-0,993%	-1,597%	-0,999%	-0,401%	+0,736%	-2,995%	+0,335%

Analisando a Tabela 5, nota-se que a ALNS gerou os melhores resultados para os problemas da segunda-feira, terça-feira, quarta-feira, quinta-feira e sábado. Para a sexta-feira e domingo, as soluções do ALNS foram, até 0,7% piores que as melhores soluções geradas.

Analisando a qualidade das soluções do ALNS nos quatro cenários, apenas o problema do sábado, nos Cenários $1TV_{600}$ e $2TV_{600}$ apresentou uma quantidade de jornadas com dupla pegada superior aos 20%, indesejável pelas empresas. As demais soluções contam com o número de duplas pegadas inferior a 20% das jornadas. Outro fator a ser notado é que nos quatro cenários estudados, a ALNS proposta superou a solução adotada pela empresa em todos os dias analisados.

Tabela 6. Comparação do custo das melhores soluções obtidas nos cenários considerados

	$1TV_{600}$	$2TV_{600}$	$1TV_{5.000}$	$2TV_{5.000}$
Segunda	1.211.836	**1.203.568**	1.261.280	**1.257.688**
Terça	1.169.944	**1.153.616**	1.201.956	**1.197.704**
Quarta	**1.384.688**	1.388.460	**1.431.336**	1.439.944
Quinta	1.514.668	**1.513.188**	**1.564.068**	1.575.324
Sexta	**1.431.040**	1.431.724	1.475.376	**1.469.816**
Sábado	**1.053.000**	1.055.320	1.126.256	**1.117.452**
Domingo	565.224	**564.696**	599.212	**594.592**

Na Tabela 6 é apresentado um resumo dos resultados obtidos nos diferentes cenários. Considerando os resultados associados a V_{600}, nota-se que a flexibilidade com as trocas de veículos não levou a um impacto significativo no custo das soluções. Para três dentre os sete problemas, foram obtidas soluções mais baratas ao permitir até uma troca de veículo por jornada, enquanto para os outros problemas, se sobressaíram as soluções que permitem realizar até duas trocas. Este fato pode ser explicado pela flexibilidade que o menor peso para as duplas pegadas proporciona, uma vez que, dado o baixo valor deste tipo de jornadas, seu número tende a ser maior. Ao analisar o segundo bloco (V_{5000}) nota-se a influência de se realizar mais trocas de veículos no valor da FO. Neste caso, apenas dois problemas apresentaram soluções melhores com até uma troca de veículo por jornada. Como o valor das jornadas de dupla pegada é mais elevado, a busca tende a explorar situações menos custosas, executando mais trocas de veículos.

5. Conclusões

Este trabalho apresenta uma implementação da heurística *Adaptive Large Neighborhood Search* para resolver o PPT. Algumas heurísticas de remoção e inserção foram adaptadas ao contexto do PPT, e uma nova heurística de remoção de tarefas foi proposta: a Remoção Média.

Uma série de parâmetros utilizados pela ALNS foi calibrada e a melhor configuração foi selecionada, a partir de testes estatísticos, para a resolução dos problemas pertecentes ao PPT em

futuras execuções. Nas instâncias de teste, a ALNS foi capaz de gerar bons resultados e superou as abordagens mais competitivas na maioria dos problemas. Foram geradas, no mínimo, 71% e, no máximo, 86% das melhores soluções ao considerar o conjunto dos 7 problemas resolvidos.

Conclui-se portanto que a ALNS é uma eficiente metodologia de resolução de problemas do tipo do PPT. Sua superioridade ao resolver estes problemas pode ser explicada pela sua característica adaptativa, o que a torna mais robusta e adaptável a intâncias de diferentes características.

Agradecimentos

Os autores agradecem à UFOP, à FAPEMIG e ao CNPq pelo apoio recebido para a realização deste trabalho.

Referências

Ahuja, R. K., Ergun, Ö., Orlin, J. B. e Punnen, A. P. (2002). A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, *123*(1-3), 75–102.

Bodin, L., Golden, B., Assad, A. e Ball, M. (1983), Routing and scheduling of vehicles and crews: The state of the art. *Computers and Operations Research*, 10, 63-211.

Chen, S. e Shen, Y. (2013), An improved column generation algorithm for crew scheduling problems. *Journal of Information and Computational Science*, *10*(1), 175–183.

Fischetti, M., Martello, S. e Toth, P. (1987) The fixed job schedule problem with spread-time constraints. *Operations Research*, *35*(6), 849–858.

Lourenço, H. R., Paixão, J. P. e Portugal, R. (2001), Multiobjective metaheuristics for the bus driver scheduling problem. *Transportation science*, INFORMS, *35*(3), 331–343.

Pisinger, D. e Ropke, S. (2007), A general heuristic for vehicle routing problems. *Computers & operations research*, Elsevier, *34*(8), 2403–2435.

Potvin, J.-Y. e Rousseau, J.-M. (1993), A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, Elsevier, *66*(3), 331–340.

Reis, A. S., Silva, G. (2012). Um Estudo de Diferentes Métodos de Busca e a Metaheurística VNS para Otimizar a Escala de Motoristas de Ônibus Urbano. *Transporte em Transformação XVI* - Trabalhos vencedores do Prêmio CNT Produção Acadêmica.

Ropke, S. e Pisinger, D. (2006a), An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, Informs, *40*(4), 455–472.

Ropke, S. e Pisinger, D. (2006b), An unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operational Research*, Elsevier, *171*(3), 750–775.

Silva, G. P. e Cunha, C. B. (2010). Uso da Técnica de Busca em Vizinhança de Grande Porte para a Programação da Escala de Motoristas de Ônibus Urbano. *Transportes*, 18, 64-75.

Silva, G. P., Souza, M. J. F. e von Atzingen, J. (2006), Métodos exatos para resolver o problema de programação da tripulação. *Transportes*, *14*(1), p. 25-32.

Silva, T. A. e Silva, G. P. (2015), O uso da metaheurística *Guided Local Search* para resolver o Problema de Escala de Motoristas de Ônibus Urbano. *Transportes*, 23, 105-116.

Song, C., Guan, W., Ma, J. e Liu, T. (2015), Improved genetic algorithm with gene recombination for bus crew-scheduling problem. *Mathematical Problems in Engineering*, 2015, 1. Hindawi Publishing Corporation.

Souza, D. S. (2014), Uma Abordagem Híbrida para Resolver o Problema da Escala de Motoristas de Ônibus Urbano. Dissertação de Mestrado – Universidade Federal de Ouro Preto.

Wren, A. (2004) Scheduling vehicles and their drivers – Forty years' experience. Technical Report, School of Computing Studies, Leeds University, England.