

## COMPARAÇÃO DE ALGORITMOS HEURÍSTICOS PARA UM PROBLEMA DE PLANEJAMENTO OPERACIONAL DE TRANSPORTE PÚBLICO

Douglas Baroni Rizzato<sup>1</sup>, Rubens Zenko Sakiyama<sup>2</sup>, Ademir Aparecido Constantino<sup>3</sup>, Wesley Romão<sup>4</sup>

Departamento de Informática – Universidade Estadual de Maringá (UEM)

Avenida Colombo, 5790 – 87020-900 – Bloco C56 – PR – Brasil

<sup>1</sup>dbr\_89@hotmail.com, <sup>2</sup>rubens.uem@gmail.com, <sup>3</sup>aaconstantino@uem.br, <sup>4</sup>wesley.uem@gmail.com

### RESUMO

Este trabalho aborda um problema de escalonamento de motoristas de uma empresa de transporte público, baseado em um caso real de grande porte, o qual tem como objetivo formar jornadas diárias de trabalho que atenda a uma série de restrições e minimize o custo operacional. Dada a complexidade computacional e escala dos dados, este trabalho propõe um algoritmo heurístico para resolver tal problema buscando minimizar os custos da solução. O algoritmo é baseado na resolução de sucessivos problemas de atribuição, sendo dividido em duas fases. Na fase da construção da solução inicial é gerado um conjunto de jornadas divididas em várias camadas. Na fase de melhoramento são empregados dois procedimentos, PCR e K-Swap, combinados com a metaheurística VND, para minimização do custo da solução inicial. Para validar o algoritmo foram realizados testes com instâncias de dados reais e aleatórias, incluindo uma instância real com mais de 2300 viagens. Em geral, os resultados alcançados obtiveram custos menores que os utilizados para comparação, evidenciando os benefícios das técnicas apresentadas.

**PALAVRAS CHAVE.** Problema de Grande Porte, Problema de Escalonamento de Motoristas, Heurística, Problema de Atribuição.

**Área principal:** Logística e Transporte, Metaheurística

### ABSTRACT

This paper tackles a bus-driver scheduling problem for a public transport company, based on large-scale real-world instances, which aims to construct a set of driver work duties satisfying a set of constraints and to minimize the operational cost. Due to computational complexity and size of the database, this work proposes a heuristic algorithm to solve the problem trying to minimize the solution cost. The algorithm is based on solving successive assignment problem which is split in two phases. In the initial construction phase it is generated a set of duties divided into several layers. In the improvement phase two procedures are used, PCR and K-swap, combined with the VND metaheuristic, to minimize the cost of the initial solution. To validate the algorithm, tests were performed using real and random instances, including real instances with more than 2300 trips. In general, the achieved results had lower costs than those used for comparison, showing the benefits of the presented techniques.

**KEYWORDS.** Large-scale instances, Drivers Scheduling Problem, Heuristic, Assignment Problem.

**Main area:** Logistics and Transport, Metaheuristics

## 1. INTRODUÇÃO

Os veículos e sua tripulação são os principais recursos de uma companhia de transporte público, e a forma como são alocados tem impacto direto na qualidade e no custo do serviço (Silva e Cunha, 2010; Silva *et al.*, 2002; Prata, 2010). Além da redução de custos direta, há o benefício proporcionado aos funcionários, que poderiam cumprir jornadas, menos exaustivas, aumentando a qualidade do serviço. Além disso, com jornadas de trabalho dentro da legislação evitam-se possíveis custos passivos (futuros) devido a ações trabalhistas (Calvi, 2005).

O Problema de Escalonamento de Motoristas (PEM), também conhecido como Problema de Escalonamento de Tripulação (PET) ou Problema de Programação da Tripulação (PPT), consiste em determinar a melhor programação diária da tripulação de forma a obedecer às restrições, minimizar custos operacionais e cumprir a escala de veículos para a tabela pré-determinada de horários. As restrições impostas a este problema são provenientes da legislação trabalhista, de acordos com o sindicato da categoria e políticas da empresa de transporte.

Devido ao grande número de combinações possíveis para a resolução do PEM e das diversas restrições a serem consideradas, o problema é de difícil solução computacional e pertence à classe NP-Difícil (De Leone *et al.*, 2010). Aliado a este fato, este trabalho utiliza uma instância de dados reais com mais de 2300 viagens ou tarefas (percurso entre dois possíveis pontos de troca da tripulação) a qual não podem ser divididas em instâncias menores. No melhor do nosso conhecimento, essa instância é a maior encontrada na literatura. Apesar de encontrarmos trabalhos que utilizam técnicas de programação matemática, como Yunes *et al.* (2005), que utilizou uma técnica baseada em geração de colunas, as instâncias utilizadas não ultrapassam de algumas centenas de viagens, onde foram utilizadas duas instâncias reais, sendo uma com 125 e outra com 246 viagens. O uso de métodos exatos ainda é um fator crítico para instâncias de grande porte. Sendo assim, este trabalho optou por investigar uma abordagem heurística para tratar o problema.

Para a resolução de pequenas instâncias do PEM (19 à 138 viagens), Santos (2007) apresentam um método exato de geração de colunas que utiliza um Algoritmo Genético (AG) para a resolução dos subproblemas de geração de novas colunas, geralmente resolvidos por PL ou por PLI.

De Leone *et al.* (2010) propõem nova formulação matemática para o PEM considerando as restrições enfrentadas por empresas de transporte italianas. Para instâncias de pequeno e médio porte, a solução exata do modelo matemático proposto foi obtida utilizando as ferramentas GAMS (*General Algebraic Modelling System*) 2005 e Cplex 9.1.2. Para grandes instâncias foi proposto um algoritmo baseado na heurística GRASP embora a maior instância não passa de 400 viagens. Silva (2010) também apresenta um modelo de resolução do PEM baseado na heurística GRASP, entretanto, a busca local é realizada pela técnica de Busca em Vizinhança de Grande Porte (BVGP). De Groot e Huisman (2004) também apresentaram uma proposta que combina algoritmo heurístico com programação matemática para solucionar instâncias reais de até 653 viagens.

Uma abordagem integrada (ou combinada) do problema de escalonamento de veículos e de motoristas para diversas garagens também é apresentada por Huisman *et al.* (2005), no qual são propostos dois diferentes modelos e algoritmos. Ambos os algoritmos são baseados em uma combinação de técnicas de geração de colunas com relaxação Lagrangeana, resolvendo instâncias com 194 à 653 viagens. Laurente e Hao (2008) também resolveram um problema integrado usando GRASP sobre instâncias reais com até 249 viagens.

Na literatura encontramos diversas propostas de algoritmos heurísticos para o PEM. Lourenço *et al.* (2001) propõem meta-heurísticas multiobjetivos usando Busca Tabu (BT) Multiobjetivo e Algoritmos Genéticos (AG) Multiobjetivo, combinados com GRASP (*Greedy Randomized Adaptive Search Procedure*). Dias *et al.* (2002) propõe um Algoritmo Genético que trata custos, número de jornadas, porcentagem da escala realizada por jornadas viáveis e duração média das jornadas como características importantes para obter uma solução de qualidade. Silva *et al.* (2002) apresentam um algoritmo baseado na meta-heurística *Simulated Annealing* (SA). Souza *et al.* (2003) fazem uma continuidade de Silva *et al.* (2002), utilizando novas estruturas de

vizinhança para o SA e para o Método VNS, que produziram soluções de qualidade e com redução do número de tripulações e horas extras quando comparadas com a solução adotada pela empresa. Silva *et al.* (2004) apresentam uma metodologia que formula o PEM como um modelo de particionamento utilizando o método *Simplex* para resolvê-lo. Para validar o método proposto foram realizados testes com instâncias reais (11 linhas contendo 11 à 87 viagens), resolvendo um problema de particionamento para cada linha separadamente, com o uso do pacote LINGO.

Mauri e Lorena (2004) propõem uma metodologia interativa baseada na aplicação do Algoritmo de Treinamento Populacional (ATP) juntamente com Programação Linear (PL) para resolver o PEM. Marinho *et al.* (2004) apresentam uma proposta baseada em Busca Tabu (BT). Gonçalves (2010) apresenta uma solução baseada em *Iterated Local Search* (ILS) e Busca Tabu.

Calvi (2005) propõe um novo algoritmo heurístico para a resolução do PEM baseado na resolução de sucessivos Problemas de Designação, também conhecido como Problema de Atribuição, que surgem de um modelo baseado em grafo multipartido. Este trabalho apresenta uma extensão dessa metodologia, apresentando um novo procedimento de busca local chamado *k-swap* e investiga sua combinação com a meta-heurística VND (*Variable Neighborhood Descent*).

Este artigo está dividido em 4 seções. A Seção 1 apresenta uma introdução ao problema estudado e faz uma revisão de trabalhos relacionados da literatura. A Seção 2 apresenta o algoritmo proposto. Na Seção 3 são apresentados os resultados computacionais. Por fim, as conclusões são apresentadas na Seção 4.

## 2. Algoritmo Proposto

Para a resolução do Problema de Escalonamento de Motoristas é proposto um algoritmo heurístico híbrido baseado na resolução de sucessivos Problemas de Atribuição (PA). O algoritmo proposto é dividido em duas fases: a construção da solução inicial e de seu melhoramento, onde faremos o uso das técnicas PCR (Procedimento de Cortes e Recombinações) e *K-Swap*.

### 2.1 Problema de Atribuição

O Problema de Atribuição (PA), também encontrado na literatura como Problema de Designação ou *Assignment Problem*, é um clássico problema de Otimização Combinatória em Pesquisa Operacional. De acordo com Hillier e Lieberman (2010), é um tipo especial de problema de programação linear no qual os designados são indicados para a realização de tarefas, que em nosso trabalho são as viagens realizadas pelas tripulações.

A solução do PA equivale ao emparelhamento perfeito de custo mínimo em um grafo bipartido. Assim, dada uma matriz de custos de dimensões  $n \times n$ , o problema consiste em associar cada linha  $i$  a uma coluna  $j$  sob um custo  $c_{ij}$ , de modo que a soma dos custos seja a menor possível. O problema pode ser formulado da seguinte maneira:

$$\text{Minimizar } \sum_{i=1}^n \sum_{j=1}^n c_{ij} \cdot x_{ij}; \quad (1)$$

$$\text{Sujeito a: } \sum_{i=1}^n x_{ij} = 1; \quad j = 1, \dots, n; \quad (2)$$

$$\sum_{j=1}^n x_{ij} = 1; \quad i = 1, \dots, n; \quad (3)$$

$$x_{ij} \in \{0,1\}; \quad i = 1, \dots, n; \quad j = 1, \dots, n. \quad (4)$$

A função-objetivo (1) minimiza o custo da soma das atribuições entre linhas e colunas; a restrição (2) exige que para cada linha haja uma coluna associada; a restrição (3) garante que

para cada coluna seja designada uma linha; a restrição (4) garante que as variáveis envolvidas assumam apenas os valores de decisão 0 e 1.

Como será detalhado nas próximas seções, o problema de designação tem diferentes interpretações para sua matriz de custo. Embora seja um problema que tenha solução em tempo polinomial isso não significa que o presente problema PEM tenha solução em tempo polinomial, pois o problema designação é apenas um subproblema que é resolvido pelo algoritmo heurístico proposto.

## 2.2 Construção da solução inicial

Inicialmente distribuimos todas as viagens em subgrupos de viagens, denominado camadas, de forma que cada camada contenha apenas viagens que não podem ser designadas em sequência. Essas camadas são utilizadas durante toda a resolução, onde serão designadas as viagens para cada motorista na formação de sua jornada (sequência de viagens realizadas por uma mesma tripulação) de trabalho e realizadas as recombinações entre partes ou pedaços de jornadas na fase de melhoramento, através da resolução de PAs.

A construção da solução inicial consiste em gerar um grafo multipartido através da resolução de sucessivos PAs, no qual cada jornada de trabalho corresponde a um caminho da primeira a última camada. Em cada camada é criada uma matriz de custos quadrada  $C^k = [c_{ij}^k]$  de ordem  $n_k$ , onde  $n_k$  corresponde a soma do total de jornadas já criadas,  $n_{jor}$ , com o total de viagens que devem ser designadas na camada  $k$ ,  $v_k$ . Na criação da matriz de custos é necessária a inclusão de viagens e/ou jornadas fictícias para que a matriz se torne quadrada, com o número de linhas igual ao número de colunas, o que é um requisito para resolver o PA, conforme alerta Pentico (2007). O algoritmo inicia com o valor de  $n_{jor}$  igual a zero e conforme as viagens vão sendo alocadas o valor de  $n_{jor}$  vai sendo incrementado conforme crescem o número de jornadas.

	Viagens	Viagens Fictícias
Jornadas	<u>Bloco 1</u> $c_{ij}^k = f(i, j)$ ou $c_{ij}^k = \infty$	<u>Bloco 2</u> $c_{ij}^k = f(i, j) + CF$
	<u>Bloco 3</u> $c_{ij}^k = CNJ$	<u>Bloco 4</u> $c_{ij}^k = 0$
Jornadas Fictícias		

**Figura 1: Estrutura da matriz de custos na construção da solução inicial**

A figura 1 ilustra a matriz de custos, formada por quatro blocos, com diferentes critérios de custos, conforme descrito a seguir:

- i. Bloco 1: caso a atribuição seja viável, a função  $f(i, j)$  retorna o custo de atribuir a viagem  $j$  à jornada  $i$ , caso contrário a função retorna um custo infinito;
- ii. Bloco 2: a função  $f(i, j)$  retorna o custo de atribuir um intervalo de folga ou um tempo ocioso à jornada  $i$ . Além disso, é acrescido um custo  $CF$  que torna a atribuição de viagens fictícias menos atrativas;
- iii. Bloco 3: A função deste bloco é permitir que novas jornadas sejam criadas quando as viagens não puderem ser designadas às jornadas já existentes. Este bloco recebe um

custo  $CNJ$ , cujo valor é o custo de uma nova jornada, considerando o tempo mínimo pago;

- iv. Bloco 4: formado pelas viagens fictícias e pelas jornadas fictícias, sua função é apenas completar a matriz. Assim, este bloco recebe custo zero.

Na criação da matriz de custos foram utilizadas duas diferentes equações,  $f1$  e  $f2$ , para atribuir o custo de uma viagem  $j$  a uma jornada  $i$ :

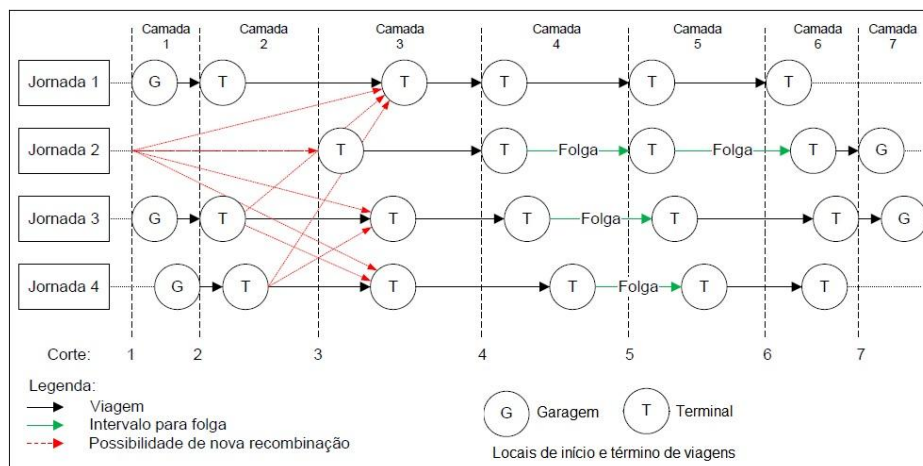
$$f1(i,j) = CO(i,j) + CHE(i,j) + CTV(i,j) + Pen.nRV \tag{5}$$

$$f2(i,j) = CO(i,j) + CTV(i,j) + Pen.nRV \tag{6}$$

Nas equações (5) e (6),  $CO(i,j)$  é o custo em minutos ociosos,  $CHE(i,j)$  é o custo de horas extras,  $CTV(i,j)$  penaliza caso haja troca de veículos,  $nRV$  é o número de restrições violadas nessa atribuição e  $Pen$  é o custo de penalidade devido às violações de restrições. A equação (5) considera o também custo de horas extras.

### 2.3 Fase de Melhoramento

A fase de melhoramento é composta por dois procedimentos, o Procedimento de Cortes e Recombinações (PCR) e o *K-Swap*, que são utilizados para minimizar o custo total da solução inicial. O PCR busca reduzir o custo total da solução através da troca de pedaços entre jornadas. Para isso, realiza um corte que divide cada uma das  $n$  jornadas em duas jornadas parciais, uma à esquerda e outra à direita do corte, conforme ilustra na Figura 2, o corte realizado entre as camadas 2 e 3.



**Figura 2: Exemplo de possíveis recombinações do PCR ao resolver a camada 3**

Após isso, é calculado o custo de se associar cada um dos  $n$  trechos de jornada à esquerda com cada um dos  $n$  trechos à direita do corte. Assim, é construída a matriz de custos  $D^k = [d_{ij}^k]$ , de dimensões  $n \times n$ , onde  $n$  corresponde ao número total de jornadas e o valor de  $d_{ij}^k$  é dado por  $g(i,j)$ , que corresponde ao custo de associar a jornada à esquerda  $i$  com a jornada à direita  $j$ , caso possível, mais as possíveis penalidades. Caso essa associação seja inviável,  $d_{ij}^k$  recebe custo infinito.

De posse da matriz de custos  $D^k$ , é resolvido o PA para a camada correspondente e com base na solução é realizada a recombinação dos trechos, podendo acarretar na formação de novas jornadas. Uma iteração completa do procedimento consiste em realizar cortes e recombinações antes de todas as camadas do problema, exceto antes da primeira camada por não resultar em nenhuma alteração da solução. A Figura 3 exemplifica como poderia ser realizada a recombinação no PCR. Note que houve uma troca entre as jornadas 3 e 4, permitindo reduzir a duração da Jornada 3.



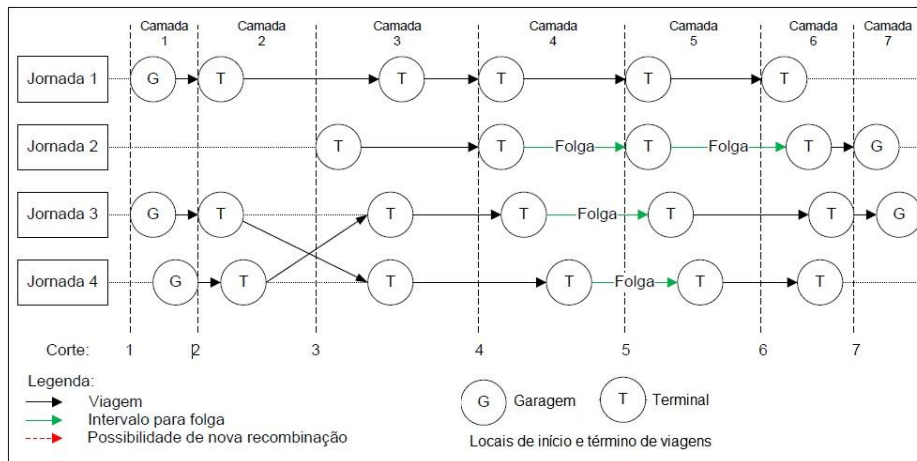


Figura 3: Exemplo de recombinação após execução do PCR na camada 3

O segundo procedimento de melhoria, o *K-Swap*, é um procedimento de trocas variáveis, e consiste em selecionar  $K$  camadas consecutivas e verificar a possibilidade de realizar trocas desses blocos de viagens entre todas as jornadas. Ou seja, o *K-Swap* busca minimizar o custo total da solução através da redistribuição de blocos de viagens entre os motoristas durante  $K$  camadas.

O *K-Swap* realiza um corte entre duas camadas, como no PCR, depois percorre  $K$  camadas e realiza um novo corte entre camadas. O trecho entre os dois cortes, denominado bloco de viagens, é a parte que poderá ser trocada entre as jornadas. De posse da variável  $K$ , é calculado o custo de se associar cada um dos  $n$  blocos de viagens com cada uma das jornadas. Assim, na matriz  $E^k = [e_{ij}^k]$ , referente ao procedimento *K-Swap*,  $e_{ij}^k$  é dado por  $h(i, j)$ , que equivale ao custo de associar o bloco de viagens  $j$  com a jornada  $i$ , caso possível, incluindo possíveis penalidades. Se a associação for inviável,  $e_{ij}^k$  recebe custo infinito.

Na Figura 4 estão representadas, em linhas pontilhadas, as possibilidades de recombinação do *1-Swap*, *K-Swap* tomando  $K$  igual a 1, ao resolver a Camada 4.

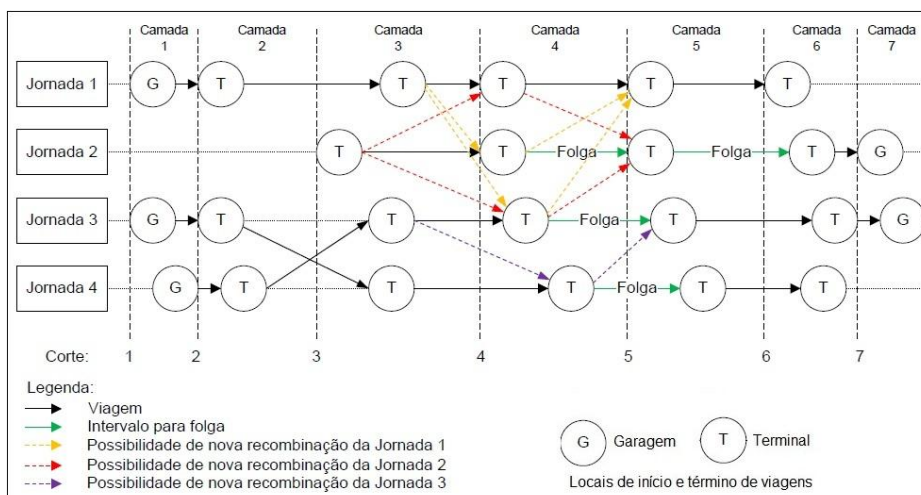
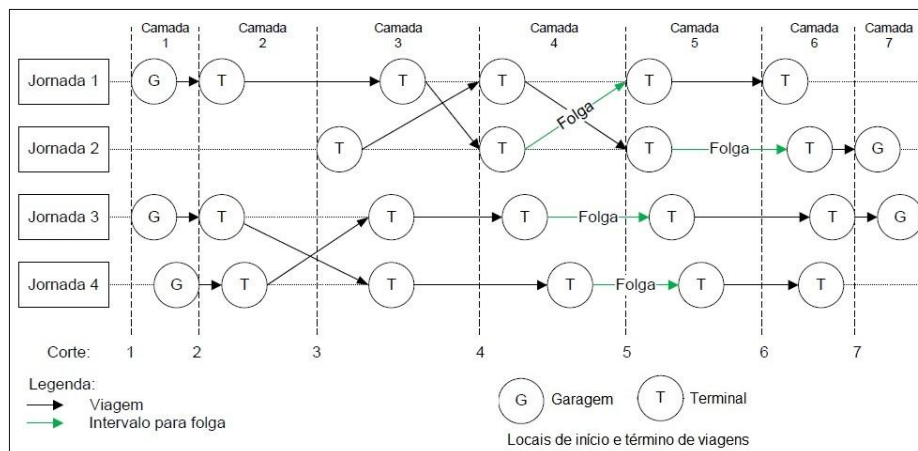


Figura 4: Exemplo de possíveis recombinações do 1-Swap ao resolver a camada 4

Construída a matriz de custos  $E^k$ , o PA para a camada correspondente é resolvido e com base na solução é feita a recombinação dos blocos de viagens, que no exemplo contém apenas uma viagem devido ao fato de  $K$  ser igual a 1. Um exemplo dessa recombinação é

apresentado na Figura 5, onde foi realizada uma troca entre os trechos das Jornadas 1 e 2, permitindo que fosse atribuído um intervalo de folga à Jornada 1.



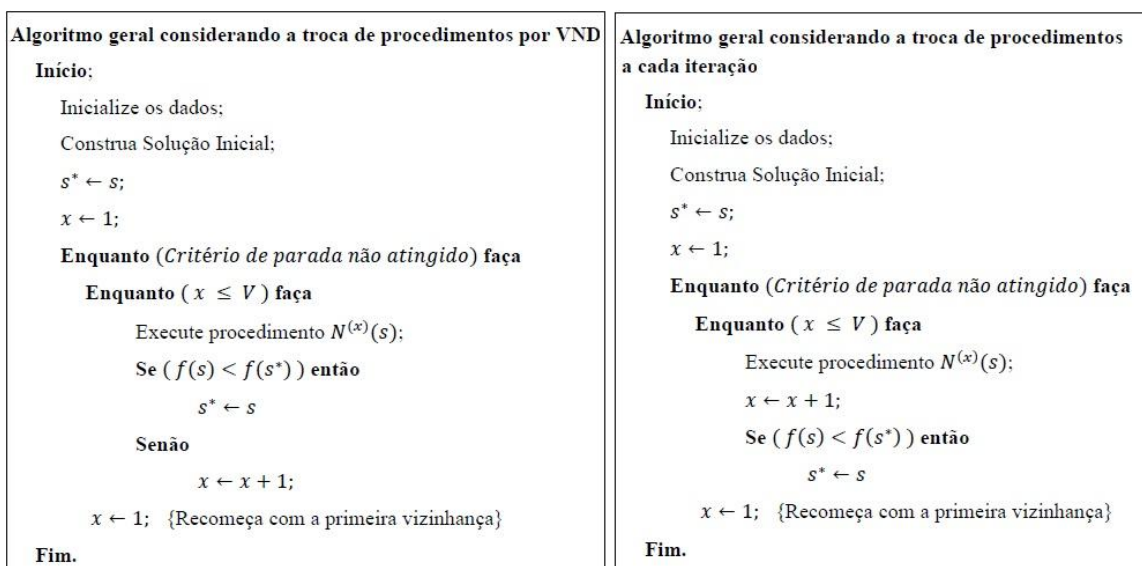
**Figura 5: Exemplo de recombinação após execução do 1-Swap na camada 4**

## 2.4 Descrição geral dos algoritmos

O algoritmo trabalha em duas fases: construção inicial e melhoramento. Na fase da construção inicial, é gerado um grafo multipartido, que representa as jornadas dos motoristas. Na fase de melhoramento, são consideradas duas abordagens.

A primeira utiliza a metodologia VND (*Variable Neighborhood Descent*), uma variação do VNS (*Variable Neighborhood Search*), que explora o espaço de soluções através de trocas sistemáticas de estruturas de vizinhança, apresentado por Hansen e Mladenovic (2001). Nesta abordagem são realizadas trocas entre os procedimentos de melhoria quando um procedimento não conseguir melhorar a solução atual. Essas trocas são realizadas de forma cíclica até que o critério de parada seja alcançado.

A outra abordagem consiste em realizar a troca de procedimentos de melhoria de forma contínua, independentemente da solução encontrada. Assim, após a execução de um procedimento sempre será executado um procedimento diferente.



**Figura 6: Descrição resumida das duas abordagens adotadas na fase de melhoramento**

As duas abordagens do algoritmo proposto, ilustrado na figura 6, seguem os mesmos critérios de parada, ou seja, 30 procedimentos/iterações, ou a execução de todos os cinco diferentes procedimentos sem a ocorrência de quaisquer melhorias na solução.

Assim, para representar as diferentes combinações dos procedimentos utilizados no algoritmo adotamos a seguinte sigla,  $P_1P_2-M_{f_l}$ , sendo:

$P_1$  e  $P_2$ : identificação do primeiro e segundo procedimento de melhoria a ser executado podendo assumir os seguintes valores:

PCR – procedimento PCR;

K(C) – procedimento K-Swap sendo  $K$  é crescente ( $K = 1, 2, 3, 4$ );

K(D) – procedimento K-Swap sendo  $K$  é decrescente ( $K = 4, 3, 2, 1$ );

$M$ : identifica se a meta-heurística VND foi utilizada, caso contrário é omitida;

$f_l$ : identifica a função de custo utilizada ( $l = 1$  ou  $2$ ), ou seja,  $f_1$  ou  $f_2$ .

### 3. Resultados

#### 3.1 Considerações iniciais

Os algoritmos propostos foram implementados utilizando a ferramenta de programação de código aberto Lazarus Pascal. Os experimentos foram realizados em uma máquina Dell Inspiron com um processador Intel Core 2 Duo de 2,53 GHz, 4 GB de memória RAM e rodando o sistema operacional Windows 7. Calvi (2005) utilizou a ferramenta de programação Delphi 4 e equipamento Pentium III de 900 MHz com 512MB de memória RAM e rodando o sistema operacional Windows 2000. Para a resolução dos problemas de atribuição foi utilizando o algoritmo de Carpaneto e Toth (1987), que combina o método húngaro com o método *shortest augmenting path*.

Foram utilizadas duas instâncias de dados reais, fornecidas por duas empresas de transporte urbano de passageiros do estado do Paraná, e oito instâncias geradas aleatoriamente a partir de uma instância real. Nestas instâncias, a escala de veículos já está definida e segue o planejamento das empresas.

Na Tabela 1 são apresentados, da segunda a quarta coluna, os melhores resultados encontrados pelos algoritmos propostos e, na quinta coluna, a sigla referente à configuração que obteve tal resultado. Da sexta até a oitava coluna são apresentados os melhores resultados de Calvi (2005). Por fim, na última coluna é apresentada uma comparação entre os melhores resultados obtidos pelos algoritmos propostos e os melhores resultados encontrados por Calvi (2005), sendo estes valores encontrados pela expressão:

$$Gap = \frac{\text{Custo algoritmo proposto} - \text{Custo Calvi (2005)}}{\text{Custo Calvi (2005)}} \cdot 100\% \quad (7)$$

Tabela 1: Comparação dos melhores resultados com os resultados de Calvi (2005)

Instância	Melhores resultados encontrados				Melhores resultados Calvi (2005)			GAP Custo
	Nº Jorn.	Custo (min)	Tempo (mm:ss)	Método	Nº Jorn.	Custo (min)	Tempo (mm:ss)	
AL130	19	8.400,93	00:18	PK(D)_F2	19	<b>8.389,40</b>	00:45	0,137%
AL251	39	<b>17.190,00</b>	00:42	K(D)P_F1	40	17.600,00	00:35	<b>-2,330%</b>
RE412	65	<b>29.260,00</b>	00:49	PK(D)_F1	66	29.512,50	00:48	<b>-0,856%</b>
AL512	76	<b>33.822,50</b>	01:28	K(C)P_F2	79	35.105,00	00:40	<b>-3,653%</b>
AL761	108	47.932,29	01:57	PK(C)_F2	107	<b>47.532,90</b>	01:50	0,840%
AL1000	144	<b>64.097,29</b>	05:17	K(C)P_F2	146	64.873,60	05:00	<b>-1,197%</b>
AL1253	182	<b>81.047,50</b>	08:45	K(D)P_F2	187	82.842,90	03:36	<b>-2,167%</b>
AL1517	221	<b>98.283,21</b>	08:52	K(D)P_F2	225	99.852,80	05:29	<b>-1,572%</b>
AL2010	285	<b>126.747,71</b>	19:27	K(D)P_F2	290	128.964,20	07:00	<b>-1,719%</b>
RE2313	327	<b>145.701,43</b>	22:30	K(D)P_F2	331	147.215,00	04:37	<b>-1,028%</b>

Com as informações da Tabela 1 constatamos que os algoritmos propostos encontraram soluções com custos menores para a maioria das instâncias, decorrente de soluções com menor



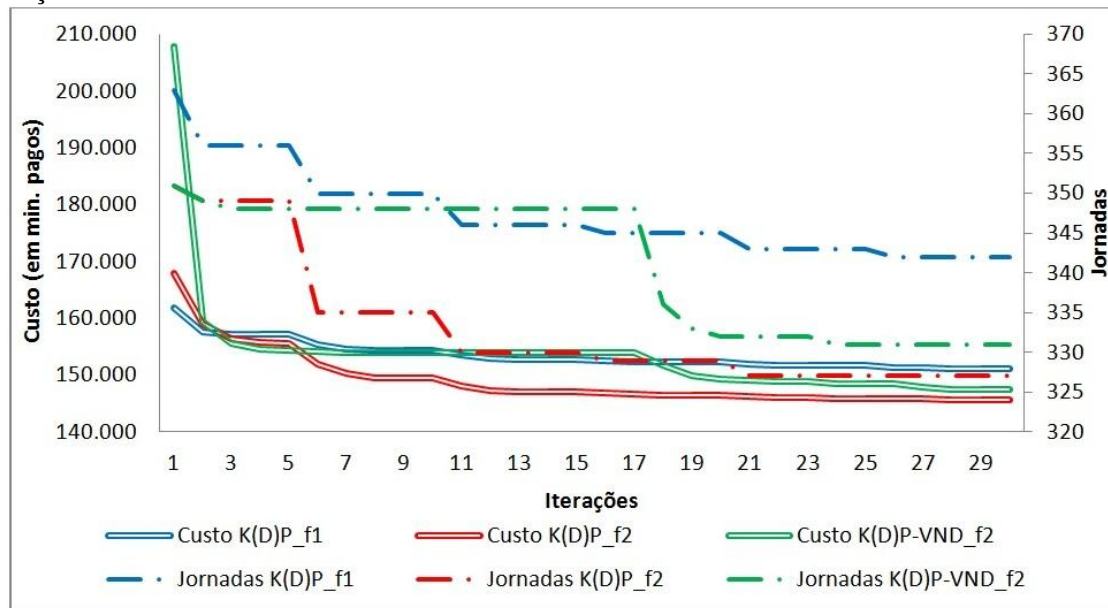
número de jornadas, exceção feita às instâncias AL130 e AL761. Em relação ao tempo de processamento, os algoritmos propostos foram executados em maior tempo, principalmente para instâncias maiores, com exceção da instância AL130.

Observando os métodos que encontraram os melhores resultados, é possível constatar que a maioria dos melhores resultados foi obtida utilizando a função  $f_2$  na construção da solução inicial e dentre os melhores resultados, nenhum foi obtido por um algoritmo que considerasse a técnica VND.

Para investigar estas constatações e os impactos que a resolução do problema sofre pela utilização das funções  $f_1$  e  $f_2$  e da utilização ou não do VND, foi gerado um gráfico de custo por iterações para a instância RE2313 utilizando os algoritmos K(D)P\_f1, K(D)P\_f2 e K(D)P-VND\_f2, apresentados na Figura 7.

No início da resolução, nem todas as restrições são satisfeitas, então para transformar estas restrições em custos, foi estabelecido um custo de penalidade  $Pen$  com valor igual ao valor de uma nova jornada, ou seja, 440 minutos pagos, que correspondem às 7 horas e 20 minutos que é o tempo mínimo pago por uma jornada normal.

Observando o gráfico da Figura 7 e comparando inicialmente os métodos K(D)P\_f1 e K(D)P\_f2, fica claro que pelo fato de não considerar custos de horas extras, a utilização da função  $f_2$  na construção da solução inicial permite encontrar uma solução com maior custo, porém com menor número de jornadas que ao utilizar a função  $f_1$ . Uma solução com menor número de jornadas possibilita os procedimentos de melhoramento a encontrarem melhores soluções.



**Figura 7: Gráfico de custo x iterações para os métodos K(D)P\_f1, K(D)P\_f2 e K(D)P-VND\_f2**

O motivo da técnica VND não ter contribuído para nenhuma das melhores soluções é fato de só ocorrer troca do procedimento de melhoria quando o procedimento em execução não conseguir mais realizar melhorias, ficando muito tempo no mesmo procedimento realizando pequenas melhorias, aumentando assim o tempo de convergência do algoritmo.

Apesar dos tempos de processamento do algoritmo proposto terem sido maiores que os de Calvi (2005), a Tabela 2 apresenta o número de jornadas, o custo, a iteração e o tempo de execução do algoritmo no momento em que um custo menor que o de Calvi (2005) é encontrado.

Pela análise da Tabela 2, é possível perceber que resultados melhores que os de Calvi (2005) são encontrados bem antes do tempo total de execução do algoritmo proposto e, em alguns casos, até mesmo com um tempo menor que o de Calvi (2005). Entretanto, pelo fato dos algoritmos terem sido executados em máquinas diferentes não é justo fazer tal comparação com relação ao tempo, visto que as máquinas atuais possuem desempenho superior.

Tabela 2: Tempo de execução quando um custo menor que o de Calvi (2005) é encontrado

Instância	Melhores resultados encontrados					Melhores resultados Calvi (2005)		
	Nº Jorn.	Custo (min)	Iteração	Tempo (mm:ss)	Método	Nº Jorn.	Custo (min)	Tempo (mm:ss)
AL130	19	8.400,93	-	-	PK(D)_F2	19	<b>8.389,40</b>	00:45
AL251	39	<b>17.347,50</b>	2	0:10	K(D)P_F1	40	17.600,00	00:35
RE412	65	<b>29.470,00</b>	27	0:45	PK(D)_F1	66	29.512,50	00:48
AL512	76	<b>34.673,00</b>	18	0:53	K(C)P_F2	79	35.105,00	00:40
AL761	108	47.932,29	-	-	PK(C)_F2	107	<b>47.532,90</b>	01:50
AL1000	144	<b>64.802,50</b>	20	3:32	K(C)P_F2	146	64.873,60	05:00
AL1253	182	<b>82.711,36</b>	8	2:20	K(D)P_F2	187	82.842,90	03:36
AL1517	221	<b>99.498,36</b>	12	3:33	K(D)P_F2	225	99.852,80	05:29
AL2010	285	<b>128.846,64</b>	12	7:47	K(D)P_F2	290	128.964,20	07:00
RE2313	327	<b>147.100,86</b>	13	9:45	K(D)P_F2	331	147.215,00	04:37

### 3.2 Comparação entre os procedimentos de melhoramento

Para comparar os procedimentos de melhoramento, foi gerada uma solução inicial utilizando a função  $f_2$  para cada uma das instâncias. A partir disso, os diferentes procedimentos de melhoramento foram empregados isoladamente seguindo os mesmos critérios de parada. No gráfico da Figura 8, é possível visualizar a redução percentual dos custos que cada procedimento foi capaz de propiciar a uma mesma solução inicial obtida. O gráfico também apresenta a redução alcançada pela combinação dos procedimentos.

Pela análise do gráfico nota-se que nenhum procedimento executado de forma isolada conseguiu atingir a mesma redução que a combinação entre eles. Nota-se também, que com exceção das duas menores instâncias, AL130 e AL251, os procedimentos que proporcionaram maiores reduções nos custos foram o PCR e o 4-Swap, sendo este último mais efetivo individualmente. Com exceção às instâncias AL130 e AL251, o procedimento que resultou na menor redução de custos foi o 1-Swap, seguido pelo 2-Swap, sendo que para as instâncias AL130 e AL251 este papel coube ao PCR.

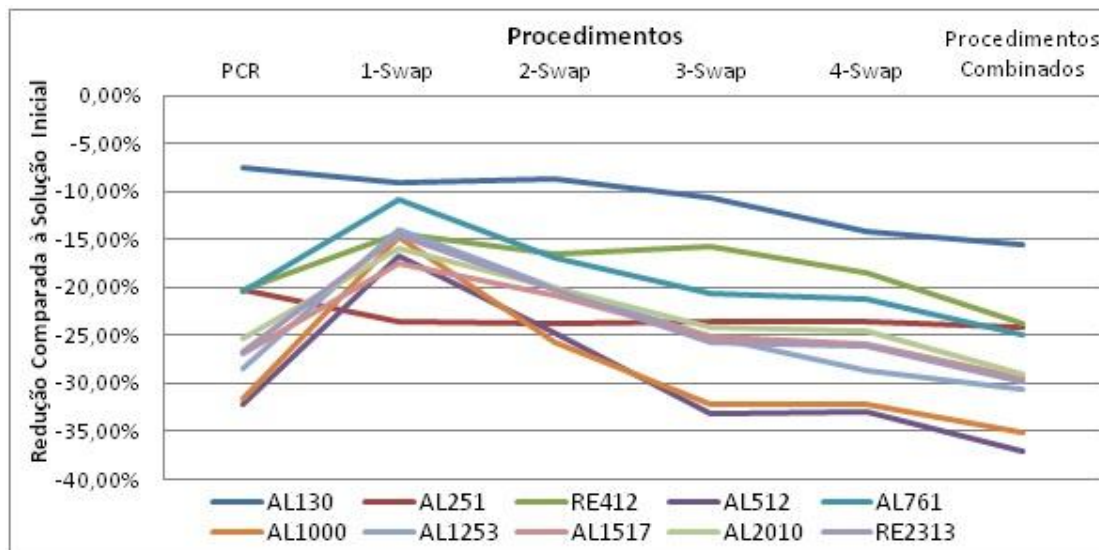


Figura 8: Redução percentual considerando os diferentes procedimentos de maneira isolada comparados com a melhor solução obtida

#### 4. Conclusão

Este trabalho apresenta uma nova abordagem heurística para o problema de escalonamento de motoristas, combinando métodos exatos (problemas de designação) com heurísticas construtivas e de busca local juntamente com VND. Os testes foram realizados com instâncias reais e fictícias com até 2313 viagens, portanto, é a maior instância do problema que temos conhecimento.

As soluções obtidas alcançaram, para a maioria dos casos, resultados computacionais melhores que os utilizados para comparação, de Calvi (2005). Conforme resultados apresentados, os algoritmos propostos conseguiram produzir boas soluções tanto para as instâncias de pequeno porte quanto para as maiores. Para ilustrar tal redução, vale ressaltar que para a maior instância, com 2313 viagens, foi obtida uma redução de 331 para 327 motoristas, representando uma economia de 1513 minutos, ou aproximadamente 25 horas por dia. Para a segunda maior instância, com 2010 viagens foi possível reduzir o número de jornadas de 290 para 285, resultando numa economia de aproximada de 37 horas de trabalho.

Estas reduções de custo obtidas correspondem ao período de apenas um dia de planejamento, sendo assim, isso pode representar uma redução de mais de 700 horas mensais, no caso com 2313 viagens. Outra vantagem, também apresentada por Calvi (2005), é a construção de escalas que atendam às regras operacionais e restrições trabalhistas, evitando insatisfações e ações trabalhistas. Desta forma, fica evidenciada a importância da PO na resolução deste tipo de problema, por permitir a resolução computacional do problema com maior eficácia e eficiência.

Comparando com a metodologia proposta por Calvi (2005), a principal diferença em relação ao método proposto refere-se ao procedimento de melhoria *K-Swap*, que permite realizar trocas entre partes de jornadas e considerando diferentes tamanhos para essas partes. Isso foi possível porque essa nova busca na vizinhança permitiu explorar um espaço de busca maior, com diferentes combinações entre jornadas.

Apesar dos tempos computacionais necessários para atingir os melhores resultados terem sido maiores em comparação aos tempos obtidos por Calvi (2005), soluções com custos ligeiramente inferiores foram obtidas em tempos similares.

Como trabalho futuro pretende-se investigar novos operadores de busca na vizinhança, por exemplo, focados na redução de custos com horas extras, e a combinação desses procedimentos em uma meta-heurística tal como VNS (*Variable Neighborhood Search*) que geralmente faz uma investigação mais ampla que VND.

**Agradecimentos:** Agradecemos a CAPES e ao CNPq pelo suporte financeiro que apoiaram o desenvolvimento deste trabalho

#### Referências

- Baptiste, P.** (2001), Combining Operations Research and Constraint Programming to Solve Real-Life Scheduling Problems. *ERCIM News*, n. 44, jan./2001.
- Calvi, R.**, *Um Algoritmo para o Problema de Escalonamento de Tripulação em Empresas de Ônibus*. 2005. Dissertação de Mestrado. Universidade Estadual de Maringá, Maringá.
- Carpaneto, G. e Toth, P.**, Primal-dual algorithms for the assignment problem. *Discrete Applied Mathematics*. 1987, 18, p.137–153.
- Constantino, A. A., Netto, C. A. S. e Araújo, S. A.** (2006), Problema de Escalonamento de Pessoal em Centrais de Atendimento Telefônico. *Anais do XXXVIII SBPO*, p. 1670-1681.
- De Leone, R., Festa, P. e Marchitto, E.** (2011), A Bus Driver Scheduling Problem: a new mathematical model and a GRASP approximate solution. *Journal of Heuristics*, vol. 17, n. 4, p. 441-466.
- De Groot S, Huisman D.** (2004), Vehicle and crew scheduling: solving large real-world instances with an integrated approach. *Econometric Institute Report EI2004-13*, Erasmus University, Rotterdam, The Netherlands; p. 43-56.

- Dias, T. G., Souza, J. P. e Cunha, J. F.** (2002), Genetic algorithms for the bus driver scheduling problem: a case study. *Journal of the Operational Research Society*, 53, p. 1-12.
- Ernst, A. T., Jiang, H., Krishnamoorthy, M. e Sier, D.** (2004), Staff Scheduling and Rostering: A review of applications, methods and models. *European Journal of Operational Research*, vol. 153, p. 3-27.
- Gonçalves, T. L.**, *Meta-heurísticas para o Problema de Programação de Tripulações*. 2010. Dissertação de Mestrado. Universidade Federal do Rio de Janeiro, Rio de Janeiro.
- Hansen, P. e Mladenovic, N.** (2001), Variable Neighborhood Search: Principles and applications. *European Journal of Operational Research*, v. 130, p. 449-467.
- Hillier, F. S. e Lieberman, G. J.**, *Introdução à Pesquisa Operacional*. 8. Ed. McGraw-Hill, Porto Alegre, 2010, p. 852.
- Huisman, D., Freling, R. e Wagelmans, A. P. M.** (2005) Multiple-Depot Integrated Vehicle and Crew Scheduling. *Transportation Science*, v. 39, n. 4, p. 491–502.
- Laurent, B. e Hao, J. K.** (2008) Simultaneous Vehicle and Crew Scheduling for Extra Urban Transports. *Lecture Notes in Artificial Intelligence* 5027: 466–475.
- Lourenço, H. R., Paixão, J. P. e Portugal, R.** (2001), Multiobjective Metaheuristics for the Bus Driver Scheduling Problem. *Transportation Science*, v. 35, n. 3, p. 331-343.
- Marinho, E. H., Ochi, L. S., Drummond, M. A., Souza, M. J. F. e Silva, G. P.**, (2001), Busca Tabu Aplicada ao Problema de Programação de Tripulações de Ônibus Urbano. *Anais do XXXVI SBPO*, p. 1471-1482.
- Mauri, G. R. e Lorena, L. A. N.** (2004), Método Interativo para Resolução do Problema de Escalonamento de Tripulações. *Anais do XXXVI SBPO*.
- Pentico, D. W.** (2007), Assignment Problems: A golden anniversary survey. *European Journal of Operational Research*, vol. 176, p. 774 – 793.
- Pinedo, M. L.**, *Theory, Algorithms and Systems*, 3. ed, Prentice Hall, p. 678, New York, 2008.
- Prata, B. A.**, Programação Integrada de veículos e motoristas: uma visão geral. *Revista Eletrônica Sistemas e Gestão*, v. 4, n. 3, p. 182 – 204, set./dez. 2010.
- Rizzato, D. B., Constantino, A. A. e Mello, E. L.** (2010), Automação e Otimização do Escalonamento de Enfermeiros. *Anais do XXX Encontro Nacional de Engenharia de Produção*, v. 1. p. 1-13.
- Santos, A. G.**, *Método de Geração de Colunas e Meta-heurísticas para Alocação de Tripulação*, 2008. Tese de Doutorado, Universidade Federal de Minas Gerais, Belo Horizonte.
- Santos, A. G. e Mateus, G. R.** (2007), Crew Scheduling Urban Problem: an Exact Column Generation Approach Improved by a Genetic Algorithm. *IEEE Congress on Evolutionary Computation*, p. 1725–1731.
- 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. *Revista Transportes*, v. 8 n. 2 p. 37-45.
- Silva, G. P., Souza, M. J. F. e Alves, J. M. C. B.** (2002), *Simulated Annealing* Aplicado à Programação da Tripulação no Sistema de Transporte Público. In *XXII Encontro Nacional de Engenharia de Produção*.
- Silva, G. P., Souza, M. J. F. e Gomes Jr., A. C.** (2005), Uma Metodologia Baseada em Emparelhamentos Sucessivos Aplicada ao Problema de Programação de Tripulações. *Anais do XXXV Simpósio Brasileiro de Pesquisa Operacional*, p. 1037-1046.
- Silva, G. P., Souza, M. J. F. e Reis, J. A.**, (2004), Um método exato para otimizar a escala de motoristas e cobreadores do sistema de transporte público. *Anais do XVIII Congresso de Pesquisa e Ensino em Transportes*, p. 340-346.
- Souza, M. J. F., Rodrigues, M. M. S., Mapa, S. M. S. e Silva, G. P.** (2003), Um estudo das heurísticas Simulated Annealing e VNS aplicadas ao problema de programação de tripulações. *XXIII Encontro Nacional de Engenharia de Produção*, 2003.
- Yunes, T.H., Moura, A. V., Souza, C.C.**, 2005. Hybrid Column Generation Approaches for Urban Transit Crew Management Problems. *Transportation Science*, 39(2), 273-288.