UM MODELO EXATO PARA RESOLVER O PROBLEMA DA ESCALA DE MOTORISTAS DE ÔNIBUS URBANO

Danilo S. Souza
Universidade Federal de Ouro Preto
Departamento de Computação – ICEB
danilo.gdc@gmail.com

Gustavo P. Silva
Universidade Federal de Ouro Preto
Departamento de Computação – ICEB
gustavo@iceb.ufop.br

RESUMO

Este trabalho apresenta um modelo exato para o Problema da Programação de Tripulações (PPT), o qual tem como objetivo designar as jornadas de trabalho para as tripulações de uma empresa de transporte público com o menor custo possível. O problema é do tipo NP-difícil devido às restrições operacionais impostas pela empresa, à legislação vigente, e aos acordos provenientes das Convenções Coletivas de Trabalho. Neste trabalho é apresentado um modelo inédito de programação linear inteira para resolver o problema. O modelo foi implementado utilizando a linguagem *Mathematical Programming Language* (MPL), e o mesmo é resolvido utilizando o *solver* CPLEX. Os resultados obtidos são apresentados de forma a verificar a eficiência do modelo para solucionar o problema respeitando as restrições apresentadas.

PALAVARAS CHAVE. Programação de Tripulações de Ônibus, Programação linear, Métodos exatos.

Área principal. L&T.

ABSTRACT

This paper presents an exact model for the Crew Scheduling Problem (CSP), which aims to designate the daily duties for the crews of a public transport company at the minimum cost. The problem is NP-hard, due to operational constraints imposed by the company, the current legislation, and agreements from the drivers union. In this paper we present a new model of integer linear programming to solve the problem. The model was implemented using the Mathematical Programming Language (MPL), and it is solved using the CPLEX solver. The results are presented in order to verify the efficiency of the model to solve the problem based on real cases.

KEYWORDS. Bus Crew Scheduling, Linear Programming, Exact Methods.

Main area. L&T , OC, PM.

1. Introdução

Empresas de transporte público estão cada vez mais buscando utilizar ferramentas computacionais para diminuir o custo de alocação de suas tripulações na frota operacional, ou seja, os motoristas e cobradores aos ônibus que realizarão as viagens diárias previamente. Neste trabalho é abordado o problema de alocação da tripulação, que consiste em atribuir tarefas às tripulações de modo a reduzir os custos desta mão de obra. Na literatura o problema é conhecido como Problema da Programação de Tripulações – PPT (*Crew Scheduling Problem*).

A importância do PPT vem do fato que o custo de mão de obra operacional de uma empresa de transporte público compõe grande parte dos gastos da mesma (Bouzada, 2002). A redução deste custo é de grande interesse das empresas, pois esse pode ser investido em outras áreas dentro da empresa, como qualificação dos seus funcionários, manutenção da frota, entre outros. Desta forma a qualidade do serviço pode ser melhorada, assim como o retorno à empresa.

Devido às restrições operacionais vigentes nas empresas e às leis trabalhistas e os acordos contidas nas convenções coletivas de trabalho o problema torna-se de grande complexidade. Este problema pode ser modelado como um problema de particionamento ou de recobrimento com variáveis binárias, se torna *NP-hard*, para o qual não existe algoritmo polinomial que obtenha a solução ótima (Fischetti *et al*., 1987).

O PPT é um problema clássico, para o qual existem diversos trabalhos que exploram diferentes técnicas de programação linear inteira. Entre elas são destacadas *Branch-and-Bound* (Fores *et al*., 1999; Smith e Wren, 1988), *Branch-and-Price* (Desrochers e Soumis. 1989; Barnhart *et al*. 1998) e *Branch-and-Cut* (Haase e Friberg, 1999). Como estes são modelos exatos, eles são capazes de resolver apenas problemas de pequeno porte. Para resolver problemas de grande porte, é necessário utilizar metaheurísticas de otimização tais como Algoritmos Genéticos, GRASP, Busca Tabu, Simulated Annealing, entre outros (Shen e Kwan, 2001; Silva e Cunha, 2010; Lourenço *et al*., 2001; Souza *et al*., 2004). Com o emprego destas metaheurísticas foi possível obter resultados muito satisfatórios para o problema, embora a otimalidade das soluções não sejam garantidas. Alguns trabalhos como (Gonçalves *et al*., 2008; Mauri e Lorena, 2007) propõe métodos híbridos onde a programação matemática e metaheurísticas são combinadas.

Algumas destas técnicas de otimização tem sido adotas em empresas nas últimas décadas. Uma heurística *run-cutting* para resolver o problema foi utilizada no pacote RUCUS (Wilhelm, 1975; Luedtke, 1985), desenvolvido nos Estados Unidos da América a partir da década de 70, e um algoritmo desenvolvido por Parker e Smith (1981) foi utilizado no sistema TRACS, o qual simulava o processo manual de alocação da tripulação. A formulação de recobrimento de conjuntos foi usada no pacote IMPACS do sistema BUSMAN e no pacote Crew-opt, posteriormente foi utilizado na versão inicial do HASTUS que substituiu o TRACS.

O PPT tem como dados de entrada a Programação dos Veículos que corresponde ao conjunto de ônibus de uma empresa em operação e o sequenciamento das viagens a serem realizadas por cada ônibus. Cada viagem a ser realizada pela empresa apresenta uma hora de início e de fim, além do seu ponto inicial e o seu ponto final. O conjunto de todas as viagens realizadas por um veículo constitui o seu *bloco de viagens.* Desta forma é possível prever o horário de partida do veículo da garagem e o seu retorno ao final do dia, assim como o detalhamento de todas as viagens e eventuais pausas realizadas pelos veículos ao longo da operação. As viagens de cada veículo são agrupadas em *tarefas* que são uma sequencia de viagens consecutivas que deve ser executada por um único condutor, uma vez que entre estas viagens não existe tempo suficiente para que ocorra a troca da tripulação. Cada *jornada* corresponde a um conjunto de tarefas que será realizado por uma tripulação ao longo de um dia. As jornadas tem que respeitar as restrições impostas pela empresa e pelas leias e convenções de trabalho. No problema abordado estão presentes dois tipos de jornadas. A *dupla pegada* é uma jornada caracterizada por um conjunto de tarefas que tem um intervalo entre duas tarefas maior ou igual a um valor estipulado, no caso de duas horas. Este tipo de jornada é utilizado para a condução dos veículos que só operam durante os horários de pico que ocorrem no início da manhã e no final do dia. O intervalo entre os dois pedaços de jornada não é contabilizado como hora trabalhada. Em uma jornada do tipo *pegada simples* os intervalos entre as tarefas são menores do que o valor estipulado para a dupla pegada. Independentemente do tipo de jornada, as horas extras correspondem ao tempo trabalhado além do tempo normal, que no caso é de seis horas e quarenta minutos.

A resolução do PPT tem como objetivo atribuir as tarefas às tripulações de modo que cada tarefa seja realizada por uma única tripulação; as restrições do problema sejam atendidas e que o custo total composto pelo número de tripulações e a quantidade de horas extras seja minimizadas. Neste trabalho é apresentado um modelo exato de otimização por recobrimento de conjuntos (*set covering*) para resolver o PPT, baseado no modelo proposto por Smith e Wren (1988). Entretanto, diferentemente dos modelos de recobrimento e de particionamento encontrados na literatura, o modelo proposto neste trabalho é composto por uma série de restrições que permite a geração de soluções factíveis e a resolução do PPT a partir das tarefas a serem realizadas e não de um conjunto de jornadas factíveis. O modelo foi testado com problemas baseados nos dados de uma empresa e se mostraram eficientes na geração das soluções, mas em um tempo elevado de processamento.

O artigo está dividido da seguinte forma: na seção 2 é apresentando o modelo de recobrimento para o PPT, e na subseção 2.1 as restrições consideradas para modelagem do problema real. Na seção 3 é apresentado o modelo implementado para gerar as jornadas e solucionar o problema. Os resultados são apresentados na seção 4e finalmente, na seção 5, são apresentadas as conclusões do trabalho.

2. O Modelo Clássico de Recobrimento para o PPT

No modelo clássico de recobrimento cada tarefa tem que ser executada por pelo menos uma das jornadas na solução. O modelo de recobrimento para o PPT tem a seguinte forma:

$$minimize \sum_{j=1}^{n} c_j\, x_j \qquad (1)$$

$$Sujeito\ a\ \sum_{j=1}^{n} a_{ij}\, x_j \ \geq 1\ ,\ i = 1, 2,\ \ .\ .\ .\ , m \qquad (2)$$

$$x_j \in \{0,1\}\ \ j = 1, 2,\ \ .\ .\ .\ , n \qquad (3)$$

O parâmetro n corresponde ao número de jornadas pré-definidas e consideradas no problema e m representa o total de tarefas a serem executadas. O coeficiente c_j corresponde ao custo da jornada j. A variável x_j assume valor 1 se a jornada j fizer parte da solução e 0 caso contrário. A expressão (1) minimiza as jornadas com seus respectivos custos. O coeficiente a_{ij} assume valor 1 se a tarefa i faz parte da jornada j e 0 caso contrário. A expressão (2) garante que cada tarefa será executada por pelo menos uma jornada e a expressão (3) determina que a variável x seja igual a 0 ou 1.

Neste modelo as n jornadas são os dados de entrada do problema. Entretanto, para problemas com muitas tarefas, o número de jornadas factíveis obtidas com a combinação destas tarefas é infinitamente grande, o que impede até mesmo a inicialização do modelo acima.

2.1. As restrições consideradas no caso real

As restrições a serem consideradas dependem das regras da empresa e das leis trabalhistas em vigor. Neste trabalho foram consideradas as restrições a seguir.

1) Dada duas tarefas, i e j realizadas consecutivamente por uma tripulação, então:
   a. o horário de início do tarefa j deve ser posterior ao final da tarefas i;
   b. o ponto final da tarefa i deve ser igual ao ponto inicial da tarefa j;
2) A duração de cada jornada não pode exceder o tempo máximo de trabalho.
3) O número de troca de veículos em uma jornada é limitado;
4) O tempo entre o final da jornada e o início da mesma no dia seguinte tem que ser superior a um valor dado.

As restrições implementadas neste trabalho são baseadas nas regras brasileiras atuais. A restrição 1) refere-se ao tempo de início de cada tarefa onde uma tarefa j só pode ser iniciada depois que a tarefa anterior i tenha terminado, e o local de partida da tarefa j tem que ser igual ao local de chegada da tarefa *i*, ou seja, a próxima tarefa de uma jornada tem que estar sempre no mesmo terminal da tarefa que terminou. A restrição 2) refere-se ao tempo total de uma jornada, onde o mesmo não pode ultrapassar o tempo máximo de trabalho, acrescido do tempo de hora extra. Neste caso o valor estipulado é de 06:40 horas de tempo de trabalho e 02:00 horas para o tempo máximo de horas extras. Segundo a restrição 3) uma tripulação só pode trocar de veículos um número determinado de vezes, no caso uma única vez. E a restrição 4) garante que uma mesma jornada possa ser repetida pela mesma tripulação em dois dias consecutivos.

3. O Modelo Proposto

Para modelar o problema foi utilizada a linguagem *Mathematical Programming Language* (Fourer *et al*, 2003) no ambiente de desenvolvimento Gusek. A modelagem é apresentada nas subseções 3.1 a 3.5, onde são descritos os parâmetros, os conjuntos, as variáveis, as restrições e a função objetivo.

3.1. Parâmetros

Os parâmetros utilizados na modelagem do problema são:

1) Conjunto de tarefas T, sendo que |T| = qt;
2) Conjunto de veículos V, sendo que |V| = qv;
3) Conjunto de jornadas J, sendo que |J| = qj;
4) Quantidade máxima de tarefas por jornada: P, inteiro maior que 0;
5) Constante M, inteiro maior que 0 tão grande quanto necessário ;
6) Tempo máximo de uma jornada de trabalho: $Temp\_Max\_J$, inteiro maior que 0;
7) Tempo máximo de horas extras: $Temp\_Max\_HE$, inteiro maior que 0;
8) Custo de uma jornada: $Custo\_J$, inteiro maior que 0;
9) Custo por hora extra: $Custo\_HE$, inteiro maior que 0;
10) Custo de tempo ocioso: $Custo\_OC$, inteiro maior que 0;
11) Custo de uma jornada do tipo dupla pegada: $Custo\_DP$, inteiro maior que 0;
12) Quantidade máxima de jornada do tipo dupla pegada: Qnt_Max_DP, inteiro positivo;
13) Quantidade máxima de trocas de veículos de uma jornada: Qnt_Max_TV, inteiro maior ou igual a 0;
14) Intervalo mínimo entre duas tarefas para ser considerado dupla pegada $Inter\_Min\_DP$, inteiro maior que 0;
15) Intervalo mínimo da tripulação durante a realização das tarefas: $Inter\_Min\_T$, inteiro maior ou igual 0;

No item 1) o parâmetro *qt* corresponde à quantidade de tarefas do problema; em 2) *qv* corresponde ao número de veículos na frota em operação e o parâmetro *qj* apresentado em 3) se refere ao número máximo de jornadas na solução. Em 4) é apresentado P que corresponde ao número máximo de tarefas alocadas a uma jornada. O parâmetro M em 5) refere-se a uma constante muito grande (*Big M*) que é utilizada como artifício na modelagem de algumas restrições. Em 6) e 7) os parâmetros $Temp\_Max\_J$ e $Temp\_Max\_HE$ correspondem respectivamente à duração máxima de uma jornada de trabalho e ao tempo máximo de horas extras. Os parâmetros $Custo\_J$, $Custo\_HE$, $Custo\_OC$ e $Custo\_DP$ apresentados em 8), 9), 10) e 11) representam os custos de jornada, hora extra, tempo ocioso e jornada do tipo dupla pegada, respectivamente. Os itens 12) e 13) apresentam os parâmetros ψ e ω cujos valores correspondem à quantidade máxima de dupla pegadas e troca de veículos. O parâmetro $Inter\_Min\_DP$, em 14), é o tempo de intervalo mínimo entre duas tarefas para ser considera um jornada do tipo dupla pegada; e $Inter\_Min\_T$ em 15) corresponde ao valor do intervalo mínimo da tripulação durante a realização das tarefas, válido somente para jornadas do tipo pegada simples.

Alguns parâmetros utilizados são estruturados como matrizes e vetores para facilitar a modelagem do problema. Esses parâmetros são apresentados abaixo:

16) Vetor de inteiros $st[1..qt]$;
17) Vetor de inteiros $et[1..qt]$;
18) Vetor de inteiros $sl[1..qt]$;

19) Vetor de inteiros *el*[1..*qt*];
20) Vetor de interios *vc*[1..*qt*];
21) Matriz *Possivel_TV*[$t_1 \in$ T, $t_2 \in$ T]. Se (vc[t_1] ≠ vc[t_2]) então *Possivel_TV*[t_1, t_2] := 1 senão *Possivel_TV*[t_1, t_2] := 0;
22) Matriz *Possivel_DP*[$t_1 \in$ T, $t_2 \in$ T]. Se (st[t_2] - et[t_1] >= $Inter\_Min\_DP$) então *Possivel_DP*[t_1, t_2]:= 1 senão *Possivel_DP*[t_1, t_2]:=0;

Em 16), 17), 18), 19) e 20) são apresentados os parâmetros *st*[*i*], *et*[*i*], *sl*[*i*], *el*[*i*] e *vc*[*i*] que correspondem ao início, fim, local de saída, local de chegada e veículo da cada tarefa $i \in$T. As matrizes de parâmetros *Possivel_TV* e *Possivel_DP* em 21) e 22) são construídas a partir dos parâmetros da entrada, sendo que *Possivel_TV*[*i*, *j*] tem como objetivo verificar se duas tarefas *i* e *j* representam uma possível troca de veículo, caso as mesmas sejam alocadas a uma mesma jornada de forma consecutiva. A matriz *Possivel_DP*[*i*, *j*] verifica se as duas tarefas *i* e *j* podem representar uma dupla pegada, caso as mesmas sejam alocadas a uma mesma jornada de forma consecutiva e com um intervalo de tempo que caracteriza este tipo de jornada.

3.2. Variáveis de decisão

Para modelar o PPT são necessárias algumas variáveis auxiliares para que sejam geradas soluções viáveis para o problema. As soluções viáveis devem satisfazer o conjunto de restrições apresentadas na próxima subseção. Assim as variáveis de decisão do modela são dadas abaixo.

- Variável binária x[$j \in$J, $t \in$ T, $p \in$ P];
- Variável binária y[$j \in$ J];
- Variável inteira *ini_J*[$j \in$ J];
- Variável inteira *fim_J*[$j \in$ J];
- Variável inteira *OC_int*[$j \in$ J];
- Variável inteira *OC_ext*[$j \in$ J];
- Variável inteira Qnt_HE[$j \in$ J];
- Variável binária *Troca_V*[$j \in$ J, $p_1 \in$ {1,...,|P|-1}, $p_2 \in$ {1,..., P}, $t_1 \in$ T, $t_2 \in$ T];
- Variável binária *Dupla_P*[$j \in$ J, $t_1 \in$ T, $t_2 \in$ T, $p_1 \in${1,...,|P|-1}, $p_2 \in$ {1,..., P}];
- Variável binária *Dupla_J*[j ∈ J];
- Variável inteira *Desconto_DP*[j ∈ J].

A variável de decisão x[*j*, *t*, *p*] recebe o valor 1 quando uma tarefa *t* é alocada à jornada *j*, na posição *p*, e 0 caso contrário. Esta variável é a principal variável de decisão, pois é a partir dela que é montado o conjunto de solução. A variável y[*j*] recebe valor 1 se a jornada *j* for incluída na solução e 0 caso contrário. As variáveis *ini_J*[*j*] e *fim_J*[*j*] recebem os valores de início e fim de uma jornada *j*. As variáveis *OC_int*[*j*] e *OC_ext*[*j*] recebem respectivamente os valores de ociosidade interna e externa de uma jornada *j*. A ociosidade interna é aquela que ocorre entre duas tarefas, caso contrário ela é uma ociosidade externa. Os valores de horas extras em uma jornada *j* são atribuídos à variável *Qnt_HE*[*j*]. A variável *Troca_V*[*j*, p_1, p_2, t_1, t_2] tem como objetivo verificar se existe uma troca de veículo em uma jornada. As variáveis *Dupla_P*[*j*, t_1, t_2, p_1, p_2] e *Dupla_J*[*j*] são utilizadas respectivamente para verificar e marcar se uma jornada é do tipo dupla pegada ou não. A variável *Desconto_DP* contém a amplitude do intervalo para jornadas do tipo dupla pegada. Esse valor é descontado da remuneração da jornada.

3.3 Função objetivo

Duas funções objetivo são propostas neste trabalho, sendo que aquela dada pela expressão (1) minimiza o número de duplas pegadas e não utilizando a restrição (R14) descrita a seguir. A outra função objetivo, expressa em (2), não minimizar o número de duplas pegadas, porém limita o número de jornadas deste tipo por meio da restrição (R14). Ambas são apresentadas abaixo e devem ser testadas separadamente. Neste trabalho foi testada apenas a função objetivo dada pela expressão (2).

Minimizar:

(1) $fo_1 = \sum_{j \in J}(\mathrm{y}[j]Custo\_J + (OC\_ext[j] + OC\_int[j])\,Custo\_OC + Qnt\_HE[j]Custo\_HE + Dupla\_J[\mathrm{j}]\;Custo\_DP)$;

(2) $fo_2 = \sum_{j \in J}(\mathrm{y}[j]Custo\_J + (OC\_ext[j] + OC\_int[j])\,Custo\_OC + Qnt\_HE[j]Custo\_HE)$;

Sendo que *Custo_J* é o custo fixo de cada jornada, *Custo_OC* é o custo de cada hora ociosa, *Custo_HE* é o custo de cada hora extra e *Custo_DP* é o custo associado a cada jornada do tipo dupla pegada, quando esta componente fizer parte da função objetivo.

3.4. Conjunto de restrições

As restrições foram consideradas e implementadas de acordo com as características descritas anteriormente para o PPT. Devido ao grande número de restrições, estas são apresentadas em blocos, respeitando suas similaridades.

(R1) $\sum_{t \in T} x[j,t,p] \le 1\,, \forall\, j \in J, p \in P$;

(R2) $\sum_{j \in J, p \in P} x[j,t,p] \ge 1\,, \forall\, t \in T$;

(R3) $\sum_{p \in P} x[j,t,p] \le 1\,, \forall\, j \in J, t \in T$;

(R4) $\sum_{t1 \in T} x[j,t_1,p+1] \le \sum_{t2 \in T} x[j,t_2,p]\,, \forall\, j \in J, p \in P-1$;

(R5) $\mathrm{y}[j] \ge \mathrm{x}[j,t,p],\ \forall\, j \in J, t \in T, p \in P$;

A restrição (R1) garante que uma tarefa seja executada por no máximo uma jornada; (R2) garante que em uma tarefa seja executada uma única vez em uma jornada; (R3) faz com que, em uma posição, seja executada somente uma tarefa. A restrição (R4) garante que as tarefas sejam alocadas às jornadas em posições contíguas, salvo a primeira posição. E a restrição (R5) verifica se a jornada esta sendo utilizada ou não, fazendo o valor da variável y[*j*] assumir o valor 1 se na jornada *j* realiza alguma tarefa alocada, e 0 caso contrário.

(R6) $\sum_{t \in T} st[t]x[j,t,1] = ini\_J[j],\ \forall\, j \in J$;

(R7) $et[t] - \mathrm{M}(1 - x[j,t,p]) \le fim_{J[j]},\ \forall\, j \in J, t \in T, p \in P$;

(R8) $Possivel\_DP[t_1,t_2] + x[j,t_1,p] + x[j,t_2,p+1] - Dupla\_P[j,t_1,t_2,p,p+1] \le 2$, $\forall\, j \in J,\ t_1 \in T,\ t_2 \in T, p \in P-1$;

(R9) $Dupla\_P[j,t_1,t_2,p,p+1] \le x[j,t_1,p],\ \forall\, j \in J, t_1 \in T, t_2 \in T, p \in P-1, t_2 > t_1$;

(R10) $Dupla\_P[j,t_1,t_2,p,p+1] \le x[j,t_2,p+1], \forall\, j \in J, t_1 \in T, t_2 \in T, p \in P-1, t_2 > t_1$;

(R11) $Dupla_{P[j,t_1,t_2,p,p+1]} = 0,\ \forall\, j \in J, t_1 \in T, t_2 \in \mathrm{T},\ \mathrm{p} \in \mathrm{P}-1$ e $Possivel\_DP[t_1,t_2] = 0$;

(R12) $\sum_{t_1 \in T, t_2 \in T, p \in P-1} Dupla\_P[j,t_1,t_2,p,p+1] \le 1, \forall\, j \in J, t_2 > t_1$;

(R13) $Dupla\_J[j] = \sum_{t_1 \in T, t_2 \in T, p \in P-1} Dupla\_P[j,t_1,t_2,p,p+1]\,, \forall\, j \in J$;

(R14) $\sum_{j \in J} Dupla\_J[\mathrm{j}] \le Qnt\_Max\_DP$;

(R15) $\sum_{t_1 \in T, t_2 \in T, p \in P-1} Dupla_{P[j,t_1,t_2,p,p+1]} * (st[t_2] - et[t_1]) = Desconto\_DP[j], \forall\, j \in J;$

(R16) $fim\_J[j] - ini\_J[j] - Desconto\_DP[j] \leq Temp\_Max\_J + Temp\_Max\_HE, \forall\, j \in J;$

(R17) $fim\_J[j] \; - \; ini\_J[j] \leq 780, \forall\, j \in J;$

As cinco restrições apresentadas inicialmente estão relacionadas à alocação das tarefas, garantindo assim que estas sejam designadas de forma coerente. As restrições (R6) e (R7) são responsáveis por atribuir os valores de $ini\_J[j]$, início da jornada, e $fim\_J[j]$, fim da jornada, que são utilizados nas outras restrições para garantir a consistência das soluções. As restrições apresentadas em R8) a R15) são utilizadas para identificar e calcular o intervalo de jornadas do tipo dupla pegada, sendo que (R8) verifica se duas tarefas são executadas de forma consecutiva, e essas podem ser de uma dupla pegada. Em caso afirmativo, a variável $Dupla\_P[j, t_1, t_2, p, p{+}1]$ recebe valor 1 indicando assim que naquela jornada haverá dupla pegada entre as tarefas t_1 e t_2, entre as posições p e p+1. As restrições (R9) e (R10) garantem que a variável $Dupla\_P$ só será ativada se as duas tarefas t_1 e t_2 foram alocadas àquela jornada. O conjunto de restrições em (R11) garante que se duas tarefas t_1 e t_2 não são candidatas a dupla pegada então a variável $Dupla\_P[j, t_1, t_2, p, p{+}1]$ nunca receberá valor 1 para estas tarefas. Nas restrições (R12), uma jornada só possa ter um intervalo de tempo do tipo dupla pegada. As restrições em (R13) verificam se existe dupla pegada em uma jornada, atribuindo o valor 1 à variável $Dupla\_J$ e 0 caso contrário. Em (R14) é limitado o número total de duplas pegadas pelo parâmetro Qnt_Max_DP e em (R15) é atribuído, como desconto de tempo, o valor do intervalo da dupla pegada à variável $Desconto\_DP$ para que possa ser descontado no cálculo da remuneração da jornada. As restrições em (R16) garantem que nenhuma jornada pode ultrapassar o tempo máximo de trabalho acrescido das horas extras. Em (R17) é garantido que o tempo entre o final da jornada e seu início no dia seguinte seja superior a 11 horas.

(R18) $OC\_ext[j] \geq Temp\_Max\_Jy[j] - (fim\_J[j] - ini\_J\,[j] - \; Desconto\_DP[j]), \forall\, j \in J;$

(R19) $OC\_int[j] \geq (fim\_J[j] - ini\_J\,[j] - \; Desconto\_DP[j]) - \sum_{t \in T, p \in P} \; (et[t] - st[t])x[j,t,p]\,, \forall\, j \in J;$

(R20) $OC\_int[j] \geq Inter\_Min\_TDesconto\_DP[j], \forall\, j \in J;$

(R21) $Qnt\_HE[j] \geq (fim\_J[j] - ini\_J\,[j] - \; Desconto\_DP[j]) - Temp\_Max\_HE, \forall\, j \in$ J;

(R22) $x[j, t_1, p] \; + x[j, t_2, p+1] \leq 2\,, \forall\, j \in J, t_1 \in T, t_2 \in T, p \in P - 1$ e $st[t_2] \geq et[t_1]$

(R23) $x[j, t_1, p] + x[j, t_2, p+1] \leq 1, \;\; \forall\, j \in J, t_1 \in T, t_2 \in T, p \in P - 1$ e $st[t_2] < et[t_1]$

(R24) $x[j, t_1, p] + x[j, t_2, p+1] - Dupla\_P[j, t_1, t_2, p, p+1] \leq 2, \forall\, j \in J, t_1 \in T, t2 \in T, p \in P - 1$ e $sl[t_2] = el[t_1];$

(R25) $x[j, t_1, p] + x[j, t_2, p+1] - Dupla_{P[j,t_1,t_2,p,p+1]} \leq 1, \;\; \forall\, j \in J, t_1 \in T, t2 \in T, p \in P - 1$ e $sl[t_2] \neq el[t_1];$

(R26) $x[j, t_1, p] + x[j, t_2, p+1] - Troca_{V[j,t_1,t_2,p,p+1]} \leq 1, \forall\, j \in J, \; t_1 \in T, t_2 \in T, p \in P - 1$ e $Possivel_{TV[t_1,t_2]} = \; 1\,;$

(R27) $Troca\_V[j, t_1, t_2, p, p+1] \leq x[j, t_1, p], \;\; \forall\, j \in J, \; t_1 \in T, t_2 \in T, p \in P - 1;$

(R28) $Troca\_V[j, t_1, t_2, p, p+1] \leq x[j, t_2, p+1], \;\; \forall\, j \in J, \; t_1 \in T, t_2 \in T, p \in P - 1;$

(R29) $\sum_{t_1 \in T, t_2 \in T, p \in P-1} Troca\_V[j, t_1, t_2, p, p+1] \leq Qnt\_Max\_TV, \forall\, j \in J;$

As restrições (R18) e (R19) calculam a ociosidade externa e interna de uma jornada. O intervalo de descanso da tripulação durante a realização das tarefas é garantido pela restrição (R20). O cálculo da quantidade de horas extras é realizado em (R21), e (R22) e (R23) são responsáveis por garantir que nenhuma tarefa possa ser iniciada antes que a anterior tenha terminado, em uma determinada jornada. Para permitir que haja troca de terminal somente em jornadas do tipo dupla pegada, as restrições (R24) e (R25) foram implementadas. Nas restrições (R26), (R27), R(28) e (R29) são responsáveis por garantir as regras de possíveis trocas de veículos.

4. Experimentos Computacionais

O modelo proposto foi testado com problemas gerados a partir de dados reais de uma empresa de transporte público que opera na região metropolitana de Belo Horzionte – MG. Os problemas foram resolvidos com o pacte computacional *solver* do CPLEX 12.5, executado em um microcomputador com processador *Intel*(R) Core(TM) i7-2600, com clock de 3.40GHz, 8 GB de memória Ram, sob platadorma Windows Seven Professional.

Foram gerados 5 problemas considerando conjutos de linhas da empresa. Os dados que caracterizam estes problemas são apresentados na Tabela 1.

Tabela 1: Apresentação dos dados

Problema	Número de Tarefas	Número de veículos	Número de variáveis	Número de restrições
1	24	2	41.320	69.355
2	33	3	124.144	224.074
3	40	4	227.280	422.331
4	45	5	344.616	658.278
5	52	6	765.600	1.462.053

Os resultados apresentados abaixo referem-se à execução do *solver* por um período máximo de 8 horas. O tempo de processamento apresentado nas Tabelas 2 a 6 refere-se ao tempo que o mesmo levou para encontrar uma solução melhor do que a anterior, no decorrer das 8 horas. As colunas "Total de horas extras" e "Total de tempo ocioso" e Tempo de processamento estão na unidade minutos.

Na Tabela 2 são apresentados os resultados para o 1º problema, onde 24 tarefas de 2 veículos são os dados de entrada. Neste caso, o tempo de processamento para encontrar a solução ótima é de 101 minutos, porém o *solver* leva aproximadamente 305 minutos para provar a otimalidade da solução encontrada.

Tabela 2: Resultado do problema 1.

Quantidade de jornadas	Total de horas extras	Total de tempo ocioso	Quantidade de duplas pegadas	Solução ótima	Gap	Tempo de processamento	Função objetivo
5	167	246	1	Não	32,1%	≈ 46	3.580
5	146	225	1	Sim	14,51%	≈ 101	3.517
5	146	225	1	Sim	0 %	≈ 305	3.517

Para o problema 2, cujos resultados se encontram na Tabela 3, a primeira solução é encontrada com cerca de 55 minutos com gap de 44,35%. Porém, a melhor solução encontrada no tempo limitado foi após 198 minutos de processamento e com gap de 38,84%. Existe a possibilidade desta solução ser a ótima, entretanto, ao final do tempo de processamento o mesmo não conseguido provar esta condição.

Tabela 3: Resultado do problema 2.

Quantidade de jornadas	Total de horas extras	Total de tempo ocioso	Quantidade de duplas pegadas	Solução ótima	Gap	Tempo de processamento	Função objetivo
8	50	491	1	Não	44,35%	≈ 55	5.391
7	249	431	1	Não	41,51%	≈ 126	5.129
7	151	403	1	Não	38,84%	≈ 198	4.905

Na Tabela 4 são apresentados os dados da resolução do problema 3, onde a melhor solução foi encontrada após aproximadamente 290 minutos. Observando o valor da função objetivo, este tem uma redução ainda considerável, mostrando assim que, com o decorrer do tempo de processamento, a solução pode ainda sofrer uma melhoria significativa.

Tabela 4: Resultado do problema 3.

Quantidade de jornadas	Total de horas extras	Total de tempo ocioso	Quantidade de duplas pegadas	Solução ótima	Gap	Tempo de processamento	Função objetivo
10	590	1.088	2	Não	94,63%	≈ 64	8.268
10	238	681	1	Não	58,08%	≈ 176	7.157
10	98	451	2	Não	54,87%	≈ 290	6.647

Na Tabela 5, são apresentados os dados da solução encontrada no decorrer do tempo. A solução encontrada com 294 minutos de processamento foi a única solução viável encontrada no intervalo, com um gap da solução ótima estimado em 65,4%.

Tabela 5: Resultado do problema 4.

Quantidade de jornadas	Total de horas extras	Total de tempo ocioso	Quantidade de duplas pegadas	Solução ótima	Gap	Tempo de processamento	Função objetivo
13	454	1046	2	Não	65,40%	≈ 294	9.754

Finalmente, na Tabela 6, é apresentado o relatório do processamento do problema 5, ocorrido no decorrer do tempo estipulado. A solução encontrada aos 54 minutos de processamento foi melhorada aos 430 minutos pela segunda solução, cujo gap estimado da solução ótima é de 67,78%. Este foi o problema de maior dimensão, cujos dados de entrada se refere a uma frota de 6 veículos e o modelo conta com 765.600 variáveis e 1.462.053 restrições.

Tabela 6: Resultado do problema 5.

Quantidade de jornadas	Total de horas extras	Total de tempo ocioso	Quantidade de duplas pegadas	Solução ótima	Gap	Tempo de processamento	Função objetivo
18	663	3.210	3	Não	94,41%	≈ 54	15.336
16	257	1.989	3	Não	67,78%	≈ 430	12.103

5. Análise dos Resultados e Conclusões

Neste trabalho foi apresentado um modelo inédito para resolver o Problema de Programação de Tripulações do Sistema de Transporte Público. O modelo se diferencia dos modelos clássicos uma vez que as jornadas são criadas durante a resolução do problema de programação linear e tem como dados de entrada as tarefas obtidas diretamente da frota em operação. Nos modelos clássicos de particionamento ou recobrimento de conjuntos, as jornadas devem ser geradas e informadas a priori para que o modelo de programação linear selecione aquelas que comporão a melhor solução. Neste processo, utilizado nos métodos clássicos, devem ser geradas jornadas promissoras dentro de um conjunto praticamente infinito de possibilidades, o que dificulta a sua resolução.

O modelo proposto foi testado resolvendo cinco problemas gerados a partir de dados reais e que contaram com frotas variando entre 2 e 6 veículos. Entre os testes realizados, apenas o para o problema com 2 veículos houve a garantia de obtenção da solução ótima. Para os demais problemas o gap estimado até a solução ótima variou entre 38,84% e 67,78%.

Neste trabalho são apresentados os resultados iniciais de um estudo de modelagem e resolução exata do PPT, o qual precisam ser melhorado para resolver problemas de grande porte. Por outro lado, o modelo se mostra interessante por apresentar uma abordagem diferenciada de um problema clássico que tem sido resolvido, principalmente, pela técnica de geração de colunas.

O trabalho deve ter sua continuidade com a obtenção de maior eficiência na resolução dos problemas maiores. Uma possibilidade é o particionamento do problema prático, dividindo a frota entre veículos que necessitam de jornadas do tipo dupla pegada e veículos que não necessitam deste tipo de jornada. Outra possível continuidade é a hibridização do método de resolução exato com heurísticas de otimização, largamente utilizadas e eficientes para resolver o problema.

Agradecimentos

Os autores agradecem ao Prof. Dr. Haroldo Gambini Santos pelos comentários e sugestões apresentadas, assim como ao CNPq, à FAPEMIG e à UFOP pelo apoio recebido no desenvolvimento deste trabalho.

Referências

Barnhart, C., Johnson, E. L., Nemhauser, G. L., et al. (1998) Branch-and-Price: Column Generation for Solving Huge Integer Programs. *Operations Research*, v. 46, n. 3, p. 316-329.
Bouzada, C. F. (2002) Análise das despesas administrativas no custo do transporte coletivo por ônibus no município de Belo Horizonte. Dissertação de Mestrado. Escola de Governo, Fundação João Pinheiro, Belo Horizonte.
Desrochers, M. e Soumis, F. (1989) A Column Generation Approach to the Urban Transit Crew Scheduling Problem. Transportation Science, v. 23, n. 1, p. 1-13.
Fischetti, M, Martello, S e P. Toth (1987). The Fixed Job Schedule Problem with Spread-Time Constraint. *Operations Research*, v. 35, p. 849-858.
Fores, S., Proll, L. e Wren, A. (1999) An Improved ILP System for Driver Scheduling. Wilson, N. H. M. (ed.) Computer-Aided Transit Scheduling, p. 43-61. Springer Verlag, Berlin, Germany.
Fourer, R., Gay, D. M. e Kernighan, B. W. (2003) *AMPL - A Modeling Language for Mathematical Programming*. Thomson Learning Academic Research Center, California, USA.
Gonçalves, T. L., Fampa, M. H. C., dos Santos, A. G. (2008) Meta-heurística Busca Tabu e Programação Matemática Uma Abordagem Hibrida Aplicada ao Problema de Programação de Tripulações. *Anais XXXI CNMAC - Congresso Nacional de Matemática Aplicada e Computacional*.
Haase, K. e Friberg, C. (1999) An exact branch and cut algorithm for the vehicle and crew scheduling problem. N. H. M. Wilson, ed. Computer-Aided Transit Scheduling, p. 63-80. Springer Verlag, Berlin, Germany.
Lourenço, H. R., Paixão, J. P. e Portugal, R. (2001) Multiobjective metaheuristics for the bus-driver scheduling problem. *Transportation Science*, v. 35, p. 331–343.
Luedtke, L. K. (1985) RUCUSII: A review of system capabilities. *Rousseau J.-M.,* p. 61-115.
Mauri, G. R. e Lorena, L. A. N. (2007) A New Hybrid Heuristic for Driver Scheduling. *International Journal of Hybrid Intelligent Systems, IOS Press,* v. 4 (1), p. 39-47. Amsterdam.
Parker, M. E. and Smith, B. M. (1981) Two approaches to computer crew scheduling. *Wren, A. (ed.), Computer scheduling of public transport, North-Holland, Amsterdam*, p. 193-221.
Shen, Y., Kwan, R. S. K. (2001) Tabu Search for driver scheduling. Computer-Aided Scheduling of Public Transport. Berlin: Springer-Velag. p. 121–135.

Silva, G. P., e da 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*, v. 18, p. 64-75.
Smith, B. M. e Wren, A. (1998) A bus crew scheduling system using a set covering formulation. *Transportation Research Part A*, v. 22, n. 2, p. 97-108.
Souza, M. J. F., Cardoso, L. X. T., Silva, G. P., Rodrigues, M. M. S., e Mapa, S. M. S. (2004) Metaheurísticas aplicadas ao Problema de Programação de Tripulações no sistema de transporte público. *TEMA Tendência em Matemática Aplicada e Computacional*, v. 5, n. 2, p. 357-368.
Wilhelm, E. B. (1975). Overview of the RUCUS package driver run cutting program (RUNS). In *Transportation Science Section, Operations Research Society of America, etc., Workshop on Automated Techniques for Scheduling of Vehicle Operators for Urban Public Transportation Services. Chicago*.