NOVOS MODELOS E ALGORITMOS PARA A PROGRAMAÇÃO INTEGRADA DE VEÍCULOS E TRIPULAÇÕES

Bruno de Athayde Prata
Centro de Tecnologia
Universidade Federal do Ceará
Teresa Galvão Dias
Departamento de Engenharia Industrial e Gestão
Faculdade de Engenharia - Universidade do Porto
Jorge Pinho de Sousa
INESC Porto, Faculdade de Engenharia
Universidade do Porto

RESUMO

A programação integrada de veículos e motoristas em transportes públicos de passageiros é um problema difícil de Otimização Combinatória, objeto de investigação continuada ao longo dos últimos anos. Diversos trabalhos têm mostrado que abordagens exatas de resolução para este problema não são úteis na prática, devido ao elevado custo computacional envolvido. Este artigo apresenta novos modelos para o problema de programação integrada de veículos e tripulantes, baseado no Problema de Máxima Cobertura. São também propostas técnicas metaheurísticas para o problema. Experimentos computacionais são apresentados e discutidos. Os resultados obtidos apontam para a possibilidade de, com o uso da abordagem proposta, se obter ganhos significativos em termos de custos de operação.

ABSTRACT

The integrated vehicle and crew scheduling problem is a hard and widely studied Combinatorial Optimization problem. Several studies have shown that exact approaches for this problem are not useful in practical situations due to the high computational costs involved. This paper describes new mathematical integrated models for vehicle and crew scheduling. These models are based on the Maximum Covering Problem formulation. In addition, metaheuristics are proposed for the problem. Computational results with real vehicle and crew scheduling problem instances are presented and discussed. These results indicate that the proposed approach has a considerable potential for achieving significant gains in terms of operation costs.

1. INTRODUÇÃO

No que concerne aos ônibus, o planejamento dos transportes urbanos pode ser decomposto nas seguintes etapas (CEDER, 2002): definição da rede de transportes, definição da tabela de horários, programação dos veículos, programação das tripulações e rotação das tripulações. A programação de veículos e de tripulações têm sido objeto de inúmeras atividades de pesquisa, devido aos ganhos significativos que podem resultar da sua otimização (DADUNA e PAIXÃO, 1995; WREN e ROSSEAU, 1995).

Por outro lado, diversos pesquisadores têm relatado a forte interação entre os problemas de escalonamento de veículos e de motoristas e os ganhos em considerá-los conjuntamente (FRIBERG e HAASE, 1999; GAFFI e NONATO, 1999; FRELING *et al.*, 1999). O problema integrado é usualmente denominado *Vehicle and Crew Scheduling Problem* – VCSP.

Vários trabalhos também relatam a dificuldade em solucionar o VCSP de forma exata (FRIBERG e HAASE, 1999; GAFFI e NONATO, 1999; FRELING *et al.*, 1999; FRELING *et al.*, 2003), fato que aconselha a adoção de técnicas heurísticas para a sua resolução.

Abordagens baseadas em métodos exatos, tais como *Branch-and-Bound*, *Branch-and-Cut*, geração de colunas e relaxações lagrangeanas são apresentadas por Friberg e Haase (1999), Gaffi e Nonato (1999), Haase *et al.* (2001), Freling *et al.* (2001), Fischetti *et al.* (2001),

Freling *et al.* (2003), Huisman (2004), Huisman *et al.* (2005), Weider (2007), Groot e Huisman (2008) e Mesquita e Paias (2008). As abordagens baseadas em métodos exatos, em alguns casos, se mostram inviáveis, devido aos tempos computacionais elevados. Outras limitações de tais abordagens consistem na elevada quantidade de parâmetros a ser ajustada.

Abordagens baseadas em heurísticas e meta-heurísticas são reportadas em Ball *et al*. (1983), Falkner e Ryan (1992), Patrikalakis e Xerocostas (1992), Wren e Gualda (1999), Valouxis e Housos (2002), Rodrigues *et al.* (2006), Laurent e Hao (2007) e Steinzen *et al.* (2007). Estas abordagens padecem de pouca generalidade, no sentido em que são muito específicas aos problemas para os quais foram desenvolvidas e de que não foram reportados experimentos em um grande número de instâncias.

O objetivo do trabalho reportado neste artigo foi elaborar novos modelos matemáticos para o VCSP, baseados no Problema de Máxima Cobertura (*Maximum Covering Problem* – MCP) e, com base nesses modelos, desenvolver novas abordagens para a sua resolução.

O trabalho é composto por mais quatro seções. Na segunda seção, são apresentadas as novas formulações propostas para o VCSP. Na terceira seção, são apresentadas as abordagens heurísticas desenvolvidas para o VCSP. Na quarta seção, são apresentados os resultados computacionais obtidos, em instâncias adaptadas da literatura e em problemas reais. Por fim, na quinta seção, apresentam-se algumas conclusões e possíveis desenvolvimentos futuros.

2. ABORDAGEM PROPOSTA: PROBLEMA DE MÁXIMA COBERTURA

2.1 Programação de veículos e tripulações sem *changeovers*

Dada uma matriz A, com $a_{ij} \in \{0,1\}$, o *Maximum Covering Problem* (MCP) consiste em cobrir a maior quantidade de linhas da matriz A com uma quantidade de colunas da matriz A menor ou igual a um valor previamente estipulado d. Neste modelo, as trocas de tripulante por veículo (*changeovers*) são proibidas, sendo que cada coluna corresponde ao serviço de um motorista. Cada linha representa um "tramo", isto é "viagem" do ônibus. Consideram-se as seguintes variáveis de decisão binárias:

$$z_i = \begin{cases} 1, & \text{se o } i\text{-ésimo tramo é coberto por um serviço presente na solução} \\ 0, & \text{caso contrário,} \end{cases}$$

$\forall\ i \in M$

$$x_j = \begin{cases} 1, & \text{se o } j\text{-ésimo serviço faz parte da solução} \\ 0, & \text{caso contrário,} \end{cases}$$

$\forall\ j \in N$

O objetivo consiste em minimizar o número de tramos descobertos. Assim, uma variante do MCP, aplicada ao VCSP sem *changeovers*, pode ser formulada como segue:

Minimizar $$z = \sum_{i=1}^{m} z_i \qquad (1)$$

Sujeito a:

$$\sum_{j=1}^{n} a_{ij} x_j + z_i = 1 \qquad \forall\ i = 1, \ldots, m. \qquad (2)$$

$$\sum_{j=1}^{n} x_j \leq d \qquad (3)$$

$$x_j \in \{0,1\} \qquad \forall\, j = 1, \ldots, n. \qquad (4)$$

$$z_i \geq 0 \qquad \forall\, i = 1, \ldots, m. \qquad (5)$$

A função objetivo (1) procura minimizar o número de tramos descobertos. Caso o *i*-ésimo tramo não possa ser coberto, o conjunto de restrições do tipo (2) implica que $z_i = 1$. Em conseqüência, tem-se que a integralidade das variáveis z_i pode ser relaxada. A restrição (3) impõe que um número máximo de *d* colunas da matriz *A* seja selecionado na solução. Note-se que as variáveis do tipo *z* podem ser consideradas como variáveis auxiliares, pois, na verdade, não se caracterizam efetivamente como decisões. Se $z_i = 1$, então ocorre um *leftover*.

2.2 Programação de veículos e tripulações com *changeovers*

Na modelagem proposta, a inovação consiste no fato que os serviços são gerados diretamente sobre as viagens e não sobre blocos de viaturas definidos previamente, como ocorre na abordagem seqüencial tradicional.

Com base em uma tabela de horários e em restrições laborais, são gerados serviços de pessoal tripulante, de modo a conceber uma matriz de serviços (matriz *A*). Com base na mesma tabela de horários, são programados os veículos para realizarem as viagens. Assim, a programação dos tripulantes não está atrelada à programação dos veículos, como ocorre na abordagem seqüencial. Ambas as programações (veículos e tripulações) são relacionadas com a tabela de horários. O objetivo seria cobrir a demanda (viagens) com os recursos disponíveis (veículos e tripulações). Tem-se, então, um *problema de cobertura com múltiplos recursos*. As variáveis de decisão (binárias) deste modelo associadas aos serviços são as definidas na seção 2.1: z_i e x_j.

As variáveis de decisão (binárias) associadas aos veículos são as seguintes:

$$y_{ik} = \begin{cases} 1, & \text{se o veículo cobre a viagem } k \text{ exactamente após a viagem } i \\ 0, & \text{caso contrário,} \end{cases}$$

$$\forall\, i,k \in M$$

Maximizar $$\sum_{i=1}^{m}\sum_{k=1}^{m} c_{ik} y_{ik} - \sum_{i=1}^{m} z_i \qquad (6)$$

Sujeito a:

$$\sum_{i=1}^{m} y_{ik} \leq 1 \qquad \forall\, k \in M \qquad (7)$$

$$\sum_{k=1}^{m} y_{ik} \leq 1 \qquad \forall\, i \in M \qquad (8)$$

$$\sum_{j=1}^{n} a_{ij} x_j + z_i = 1 \qquad \forall\, i = 1, \ldots, m. \qquad (9)$$

$$\sum_{j=1}^{n} x_j \leq d \qquad (10)$$

$x_j \in \{0,1\}$ $\forall\, j = 1, \ldots, n.$ (11)

$z_i \geq 0$ $\forall\, i = 1, \ldots, m.$ (12)

$y_{ik} \geq 0$ $\forall\, i,k \in M$ (13)

com c_{ik}=1 se $i\ \alpha\ k$, caso contrário $c_{ik} = -\infty$

A função objetivo do problema (6) é composta por duas parcelas: a primeira, que procura cobrir todas as viagens com a menor quantidade de veículos possível, e a segunda, que se destina a minimizar a quantidade de tramos descobertos por serviços de tripulação. Os conjuntos de restrições (7) e (8) impõem que um veículo cubra uma viagem apenas uma vez. Visto que a matriz de coeficientes das restrições de um problema de alocação é unimodular, a integralidade das variáveis y_{ik} pode ser relaxada. Caso o *i*-ésimo tramo não possa ser coberto, o conjunto de restrições do tipo (9) implica que $z_i = 1$. Em conseqüência desta restrição, tem-se que a integralidade das variáveis z_i pode ser relaxada. A restrição (10) impõe que um número máximo *d* de colunas da matriz *A* seja selecionado na solução. Os conjuntos de restrições (11), (12) e (13) dizem respeito à definição das variáveis do modelo.

3. META-HEURÍSTICAS PARA O VCSP

3.1 GRASP

Procurando obter um procedimento eficiente e flexível para resolver o VCSP sem *changeovers*, desenvolveu-se uma abordagem meta-heurística do tipo GRASP, implementada em MATLAB (vide Quadro 1).

Quadro 1: Heurística GRASP para a variante do MCP.

```
Parâmetros: α, num_iter, num_sol
enquanto i ≤ num_iter faça
      Fase de construção
      Algoritmo de melhoria
      Atualiza melhor solução encontrada
      i ← i +1
fim-do-enquanto
```

Para a fase de construção, a heurística clássica de Chvátal, procedimento desenvolvido para o *Set Covering Problem* (SCP), foi adaptada ao MCP. Para a fase de busca local, foi adaptado o algoritmo de melhoria de Beasley e Chu (1998) para o *Set Partitioning Problem* (SPP). O parâmetro α que regula a construção gulosa aleatorizada da meta-heurística GRASP foi calibrado de forma reativa, tal como proposto por Prais e Ribeiro (2000).

No que concerne ao número de iterações GRASP, este parâmetro, apesar de impactar a qualidade das soluções geradas, não é de difícil calibração. No sentido de gerir o *trade-off* entre qualidade da solução gerada e tempo de processamento, adotou-se um valor *num_iter* = 200.

O parâmetro *num_sol* diz respeito ao número de vezes que é executado o algoritmo de melhoria. Em outras palavras, *num_sol* representa a quantidade de soluções vizinhas pesquisadas no processo de busca local. Após experimentos, adotou-se um valor *num_sol* = *m*, pois se verificou que valores elevados de *num_sol* incorriam em maiores tempos de processamento e não traziam benefícios proporcionais a esse acréscimo de tempo.

3.2 GRASP / Algoritmo Genético

Foi desenvolvida uma abordagem heurística para o VCSP com *changeovers*. Tendo em vista o grande número de variáveis do problema, a abordagem heurística proposta segue o princípio da decomposição desenvolvido por Steinzen *et al.* (2007), segundo o qual o *vehicle scheduling* e o *crew scheduling* são decompostos, mas há uma concatenação entre os dois problemas.

Para a programação de veículos, foi desenvolvida uma heurística GRASP (vide Quadro 2). A fase de construção é composta por uma heurística onde uma função gulosa Γ é usada na inserção de viagens em cada veículo. Para cada veículo, calcula-se a diferença entre o tempo de término da última viagem alocada naquele veículo e as demais viagens.

A fase de melhoria é composta por dois tipos de movimentos: movimento de inserção caudal e movimento de inserção de blocos. O movimento de inserção caudal consiste na realocação das últimas viagens de um bloco para outro, de modo a equilibrar as viagens por veículo. O movimento de inserção de blocos tem como meta equilibrar as quantidades de viagens alocadas entre os veículos, por meio da inserção de subconjuntos de viagens (blocos) dos veículos mais utilizados para veículos menos utilizados.

Quadro 2: Heurística GRASP para a programação de veículos.

```
Parâmetros: num_iter
Fase de construção
Movimento de inserção caudal
i ← 0
enquanto i ≤ num_iter faça
      Movimento de inserção de blocos
      i ← i +1
fim-do-enquanto
```

Para a resolução do *crew scheduling*, foi desenvolvido um Algoritmo Genético (AG) híbrido (vide Quadro 3). O AG proposto consiste em uma adaptação do algoritmo proposto por Beasley e Chu (1998) à formulação de máxima cobertura. A codificação adotada para o AG foi a binária.

Quadro 3: Algoritmo Genético proposto.

```
Passo 1: Gerar aleatoriamente uma população de soluções viáveis.
Passo 2: Avaliar as soluções geradas.
Enquanto geração atual ≤ número máximo de gerações, faça
      Passo 3: Efetuar seleção por torneio.
      Passo 4: Realizar o cruzamento uniforme com probabilidade p_c=100%.
      Passo 5: Realizar mutação da prole com probabilidade p_m=100%.
      Passo 6: Aplicar os procedimentos DROP e ADD.
      Passo 7: Avaliar a aptidão da solução gerada.
      Passo 8: Substituir o pior indivíduo da população pela solução
      gerada.
Fim-do-enquanto
Imprimir a melhor solução obtida
```

Embora os indivíduos da população inicial sejam viáveis, após as operações de recombinação e mutação, esta viabilidade certamente será transgredida. Por isso é necessário corrigir a

inviabilidade da solução, bem como melhorar a solução corrigida. Neste sentido, foram adaptados para o problema de máxima cobertura, os procedimentos de melhoria propostos para o SPP por Beasley e Chu (1998).

Para mitigar o problema da convergência prematura da população, foi desenvolvido o seguinte procedimento para reposição das soluções. Se a prole gerada tiver melhor aptidão do que o indivíduo com pior aptidão na população, a prole ingressa na população no lugar do pior indivíduo. Caso a prole gerada não tenha uma melhor aptidão do que o indivíduo de pior aptidão na população, ocorrerá a substituição com uma probabilidade de 5%.

Os parâmetros utilizados nas heurísticas propostas são os seguintes: heurística para programação de veículos: *num_iter*: $\lfloor nv/2 \rfloor$; AG: tamanho da população (*maxpop*): 100; número de gerações (*maxgen*): 10000; e probabilidade de mutação (p_m): 100%.

4. RESULTADOS COMPUTACIONAIS

4.1 Instâncias utilizadas

Foram realizados experimentos computacionais com instâncias para o VCSP sem *changeovers* e com *changeovers*. Deve-se observar a importância de se considerar estas duas variantes do VCSP, já que, de acordo com suas regras de operação, o sistema de transporte público pode operar com a permissão da troca de motoristas entre veículos, bem como com a sua proibição. Logo, a abordagem proposta pode se adequar a diferentes casos de estudo.

A seguir, na Figura 1, é apresentada uma visão esquemática dos testes computacionais. As variantes estudadas para o VCSP são sem *changeovers* (C1) e com *changeovers* (C2). As instâncias estudadas foram de três grupos: SPP (NWXX), SCP (4.XX) e *real world instances* (RWXXX). No que concerne aos métodos de resolução, foram adotados um algoritmo de *Branch-and-Bound* (B&B) disponibilizado pelo "solver" utilizado, o GRASP e o GRASP/AG desenvolvidos neste trabalho.

Figura 1: Síntese dos testes computacionais.

		NWXX	4.XX	RWXXX
variante	C1	x	x	
	C2			x
método de resolução	B&B	x	x	x
	GRASP	x	x	
	GRASP/AG			x

4.2 *Branch-and-Bound*

Para o caso do VCSP sem *changeovers*, foram adaptadas instâncias dos problemas SPP e SCP disponíveis na biblioteca de Pesquisa Operacional OR-Library (http://people.brunel.ac.uk/~mastjjb/jeb/info.html). As instâncias foram divididas em dois grupos: NWXX, para os problemas adaptados do SPP, e 4.XX, para os problemas adaptados do SCP. A conversão das instâncias dos problemas supracitados foi feita da seguinte forma: foram utilizadas as matrizes *A* de cada instância, sendo ignorados os custos inerentes às colunas. Atribuiu-se um valor do número máximo de colunas equivalente a $\lceil 0.2 \times m \rceil$.

Os experimentos foram realizados com o *software* LINGO, com o modelo dado pelas equações (1) a (5). Caso a solução ótima não fosse obtida em 3600 segundos, o solver era parado e guardava-se a melhor solução obtida. Os experimentos foram realizados em um

processador Genuine Intel 1.86 GHz com 1GB de memória RAM.

As características das instâncias são as seguintes (vide Tabelas 1 e 2): número de linhas da matriz *A* (*m*), número de colunas da matriz *A* (*n*), número máximo de colunas a serem utilizadas na solução (*d*) e densidade da matriz *A* ( ). O valor de é obtido pela divisão do número de elementos da matriz *A* iguais a 1 pelo número total de elementos de *A*. Estas informações constam nas colunas 1 a 5 das Tabelas 1 e 2. A coluna 6 apresenta o valor da solução obtida para o problema com relaxação da integralidade das variáveis *x* e a coluna 7, o tempo de processamento para resolução do problema relaxado. Na coluna 8 consta o número de iterações do algoritmo simplex para resolução do problema relaxado. Na coluna 9 está a solução obtida para o problema binário; na 10, o tempo de processamento para resolução do problema binário; na 11, o número de nós do algoritmo *Branch-and-Bound*; e na coluna 12, o numero de ierações. O *duality gap*, apresentado na coluna 13, é o desvio percentual relativo entre o *zIP* e o *zLP*.

Tabela 1: Características das instâncias e resultados do *solver* – instâncias adaptadas do SCP.

1. instância	2. *m*	3. *n*	4. *d*	5. (%)	6. zLP	7. tzLP(s)	8. número de iterações	9. zIP	10. tzIP(s)	11. # nós (B&B)	12. número de iterações	13. duality gap (%)
4.1	200	1000	40	2,04	0	1	541	24	3600	91775	11899207	2400
4.2	200	1000	40	2,03	0	0	615	23	3600	77610	9761213	2300
4.3	200	1000	40	2,03	0	1	580	23	3600	78110	10648710	2300
4.4	200	1000	40	2,04	0	0	574	24	3600	72709	9991443	2400
4.5	200	1000	40	2,00	0	1	584	23	3600	64236	8597983	2300
4.6	200	1000	40	2,07	0	0	615	22	3600	83140	10845473	2200
4.7	200	1000	40	1,99	0	1	600	22	3600	61229	9508278	2200
4.8	200	1000	40	2,05	0	0	630	18	3600	87629	13427162	1800
4.9	200	1000	40	2,01	0	0	616	23	3600	75649	9697923	2300
4.10	200	1000	40	1,98	0	0	654	21	3600	75972	10248598	2100

No que se refere à formulação proposta para o VCSP com *changeovers*, foram efetuados experimentos computacionais em instâncias reais, advindas do sistema de transporte público de Fortaleza, no Nordeste do Brasil. No sistema de transportes coletivos por ônibus de Fortaleza, *changeovers* não são permitidos. Contudo, tal restrição foi relaxada para uma avaliação teórica do modelo proposto.

Na Tabela 3, são apresentados os resultados dos experimentos computacionais para o caso em que *changeovers* são permitidos e os veículos são alojados em um único depósito. A coluna 1 consiste na identificação da instância, representada pelo número da linha. A coluna 2 apresenta o número de viagens da linha. A coluna 3, o número de serviços gerados. Na coluna 4, é ilustrado o tempo de ciclo da linha. No sistema de transporte público por ônibus de Fortaleza, este tempo varia ao longo do dia. Portanto, é apresentada a moda da variável tempo de ciclo. Na coluna 5, é apresentada a densidade da matriz de serviços gerados; na 6, a quantidade máxima de colunas a comporem a solução; na 7, o número de colunas utilizado na solução. Na coluna 8, apresenta-se o número de veículos obtido na solução ótima do modelo; na 9, consta o número de tramos descobertos; na 10, o tempo de processamento para obtenção da solução ótima; e na coluna 11, consta o número de nós do B&B. Os experimentos foram realizados em um processador Genuine Intel 1.86 GHz com 1GB de memória RAM.

Tabela 2: Características das instâncias e resultados do *solver* – instâncias adaptadas do SPP.

1. instância	2. *m*	3. *n*	4. *d*	5. (%)	6. zLP	7. tzLP(s)	8. número de iterações	9. zIP	10. tzIP(s)	11. # nós (B&B)	12. número de iterações	13. duality gap (%)
NW41	17	197	4	22,10	0	0	41	0	0	0	54	0
NW32	19	197	4	24,30	0,8	0	45	1	0	7	229	25
NW40	19	404	4	26,95	0	0	42	0	0	0	27	0
NW08	24	434	5	22,39	3	0	49	3	0	0	56	0
NW15	31	467	7	19,55	0	0	103	0	1	0	92	0
NW21	25	577	5	24,89	0	0	60	0	1	1	272	0
NW22	23	619	5	23,87	0	0	50	0	0	0	27	0
NW12	27	626	6	20,00	5	0	29	5	1	0	14	0
NW39	25	677	5	26,55	0	0	48	0	1	1	380	0
NW20	22	685	5	24,70	0	0	42	0	1	0	52	0
NW23	19	711	4	24,80	2	0	58	2	1	0	38	0
NW37	19	770	4	25,83	0	0	44	0	1	0	55	0
NW26	23	771	5	23,77	0	1	41	0	0	0	48	0
NW10	24	853	5	21,18	5	1	49	5	0	0	33	0
NW34	20	899	4	28,06	0	0	45	0	0	0	67	0
NW43	18	1072	4	25,18	0,4	1	92	1	0	0	108	150
NW42	23	1079	5	26,33	0	0	50	0	0	0	37	0
NW28	18	1210	4	39,27	0	0	43	0	1	0	25	0
NW25	20	1217	4	30,16	0	0	55	0	0	0	56	0
NW38	23	1220	5	32,33	0	1	56	0	0	0	55	0
NW27	22	1355	5	31,55	0	1	40	0	0	0	28	0
NW24	19	1366	4	33,20	0	0	74	0	1	0	201	0
NW35	23	1709	5	26,70	0	0	53	0	1	0	56	0
NW36	20	1783	4	36,90	0	0	74	0	0	0	113	0
NW29	18	2540	4	31,04	0	0	50	0	1	0	31	0
NW30	26	2653	6	29,63	0	1	61	0	0	0	23	0
NW31	26	2662	6	28,86	0	0	66	0	0	0	81	0
NW19	40	2879	8	21,88	0	0	112	0	1	0	331	0
NW33	23	3068	5	30,76	0	0	61	0	1	0	15	0
NW09	40	3103	8	16,20	3	0	417	3	1	0	505	0
NW07	36	5172	8	22,12	0	1	178	0	1	0	347	0

Tabela 3: Características das instâncias e resultados do *solver* – instâncias com *changeovers*.

1. Instância	2. *m*	3. *n*	4. *tempo de ciclo* (min)	5. (%)	6. *d*	7. | *x* |	8. # veículos	9. *z*	10. *t*(s)	11. # nós (B&B)
RW406	85	712	110	4,5	21	21	12	5	2	8
RW504	64	446	48	13,9	7	6	4	10	5	84
RW905	49	382	72	12,0	8	7	4	7	4	647
RW013	75	530	55	10,6	9	8	5	11	45	11217
RW833	80	344	84	6,1	15	14	8	10	5	252
RW015	111	882	56	7,1	14	12	7	15	137	12489
RW501	64	450	48	13,9	7	6	4	10	9	144
RW407	67	608	94	5,8	14	14	10	11	3	189
RW466	74	515	52	10,7	9	8	5	11	4	24
RW609	69	513	90	6,2	14	14	9	9	1	0
RW907	72	550	72	8,2	12	11	6	7	4	49
RW810	64	435	48	14,0	7	6	4	10	6	31
RW605/606	86	766	100	4,4	20	19	14	11	7	133
RW081	59	392	48	15,2	6	5	4	14	2	16
RW070	83	744	110	4,6	21	20	11	7	1	0
RW102	104	741	50	7,6	12	12	6	9	2	0
RW201	73	505	62	9,5	10	9	6	10	3	26
RW411	84	620	62	8,2	12	10	8	15	7	232
RW316	55	421	91	7,4	12	12	7	7	1	0
RW913	80	554	42	12,4	8	7	4	10	21	1969

4.3 GRASP

Nas Tabelas 4 e 5, são apresentadas as soluções obtidas pela heurística GRASP, bem como sua performance computacional, expressa em termos de tempo de processamento.

Tabela 4: Soluções obtidas pela heurística GRASP proposta – instâncias adaptadas do SCP.

Instância	percentual de cobertura (*c*)				*gap* (%)			tempo médio (s)
	zIP	melhor	pior	média	melhor	pior	média	
4.1	88,0	89,0	87,0	87,9	-1,0	1,0	0,1	601,3
4.2	88,5	90,0	88,5	88,7	-1,5	0,0	-0,2	634,9
4.3	88,5	88,5	88,0	88,2	0,0	0,5	0,3	623,8
4.4	88,0	88,5	87,5	88,4	-0,5	0,5	-0,4	653,4
4.5	88,5	89,5	88,0	88,7	-1,0	0,5	-0,2	618,3
4.6	89,0	89,5	87,0	87,75	-0,5	2,0	1,3	599,7
4.7	89,0	89,0	87,5	88,25	0,0	1,5	0,8	585,6
4.8	91,0	89,0	87,0	88,15	2,0	4,0	2,8	608,2
4.9	88,5	89,5	86,5	87,5	-1,0	2,0	1,0	597,7
4.10	89,5	88,0	86,0	87	1,5	3,5	2,5	612

Tabela 5: Soluções obtidas pela heurística GRASP proposta – instâncias adaptadas do SPP.

Instância	percentual de cobertura – *c* (%)				gap (%)			tempo médio (s)
	zIP	melhor	pior	média	melhor	pior	média	
NW41	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,1
NW32	94,7	94,7	94,7	94,7	0,0	0,0	0,0	0,1
NW40	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,1
NW08	87,5	87,5	83,3	84,2	0,0	4,2	3,3	20,3
NW15	100,0	100,0	100,0	100,0	0,0	0,0	0,0	6,3
NW21	100,0	100,0	96,0	98,8	0,0	4,0	1,2	9,9
NW22	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,1
NW12	81,5	81,5	81,5	81,5	0,0	0,0	0,0	0,1
NW39	100,0	100,0	100,0	100,0	0,0	0,0	0,0	1,1
NW20	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,2
NW23	89,5	89,5	78,9	83,2	0,0	10,5	6,3	11,7
NW37	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,7
NW26	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,1
NW10	79,2	79,2	75,0	75,4	0,0	4,2	3,8	39,0
NW34	100,0	100,0	100,0	100,0	0,0	0,0	0,0	1,5
NW43	94,4	94,4	94,4	94,4	0,0	0,0	0,0	1,4
NW42	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,3
NW28	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,7
NW25	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,7
NW38	100,0	100,0	100,0	100,0	0,0	0,0	0,0	5,0
NW27	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,6
NW24	100,0	100,0	100,0	100,0	0,0	0,0	0,0	8,7
NW35	100,0	100,0	100,0	100,0	0,0	0,0	0,0	8,4
NW36	100,0	100,0	100,0	100,0	0,0	0,0	0,0	4,8
NW29	100,0	100,0	100,0	100,0	0,0	0,0	0,0	4,8
NW30	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,7
NW31	100,0	100,0	100,0	100,0	0,0	0,0	0,0	25,9
NW19	100,0	100,0	100,0	100,0	0,0	0,0	0,0	9,0
NW33	100,0	100,0	100,0	100,0	0,0	0,0	0,0	0,2
NW07	100,0	100,0	100,0	100,0	0,0	0,0	0,0	97,8

Com relação ao conjunto de dados do SPP adaptados para o MCP (instâncias NWXX), pode-se observar que, para as 30 instâncias analisadas, o algoritmo GRASP conseguiu obter a solução ótima para todos os problemas (pelo menos uma vez nas 10 rodadas realizadas). Em 26 destas instâncias, o algoritmo obteve a solução ótima em todas as 10 execuções.

Com relação ao conjunto de dados do SCP adaptados para o MCP, pode-se ainda observar que, para as 10 instâncias testadas, as quais não possuem a solução ótima determinada, em duas o algoritmo obteve o mesmo valor de zIP, em duas o algoritmo obteve resultados piores do que o de zIP e em 6 instâncias, o algoritmo obteve melhores resultados do que zIP.

Tais resultados demonstram a eficácia da abordagem proposta, em termos de conseguir obter soluções de qualidade para conjuntos de dados com diferentes características (quantidades de linhas e colunas, densidades, número máximo de colunas).

Nos testes realizados nas instâncias adaptadas do SPP, em 14 dos 30 conjuntos de dados observou-se que o GRASP obteve um tempo de processamento médio inferior a 1 segundo. Nos testes efetuados com as instâncias adaptadas do SCP, o GRASP utilizou um tempo de processamento cerca de 6 vezes menor do que o método exato, e, ainda assim, conseguiu obter melhores soluções.

4.4 GRASP/AG

Conforme relatado anteriormente, o GRASP destina-se à programação dos veículos e o AG destina-se à programação de tripulações. Para cada instância, as heurísticas para programação de veículos e de tripulações foram rodadas em seqüência, sendo chamadas no mesmo bloco de execução.

Na Tabela 6 é apresentada uma comparação entre as soluções obtidas pelo AG e as soluções ótimas obtidas pelo B&B. Para as 200 execuções do AG, este obteve a solução ótima em 163 rodadas, consubstanciando em um aproveitamento de 81,5%. Em 14 das 20 instâncias analisadas, o AG conseguiu obter a solução ótima em todas as 10 execuções do algoritmo, apontando para a eficácia do mesmo.

Tabela 6: Comparação entre o método exato e o Algoritmo Genético proposto.

Problema	percentual de cobertura (*c*)				*gap* (%)			Soluções ótimas em 10 execuções do GA
	zIP	melhor	pior	média	melhor	pior	média	
RW406	94,1	94,1	91,8	92,0	0,0	2,4	2,1	1
RW504	84,4	84,4	84,4	84,4	0,0	0,0	0,0	10
RW905	85,7	85,7	85,7	85,7	0,0	0,0	0,0	10
RW13	85,3	85,3	85,3	85,3	0,0	0,0	0,0	10
RW833	87,5	87,5	87,5	87,5	0,0	0,0	0,0	10
RW015	86,5	86,5	86,5	86,5	0,0	0,0	0,0	10
RW501	84,4	84,4	84,4	84,4	0,0	0,0	0,0	10
RW407	83,6	83,6	83,6	83,6	0,0	0,0	0,0	10
RW466	85,1	85,1	75,7	84,2	0,0	9,5	0,9	10
RW609	87,0	87,0	81,2	82,9	0,0	5,8	4,1	3
RW907	90,3	90,3	83,3	88,8	0,0	6,9	1,5	3
RW810	84,4	84,4	84,4	84,4	0,0	0,0	0,0	10
RW605/606	87,2	87,2	86,0	86,9	0,0	1,2	0,3	8
RW081	76,3	76,3	76,3	76,3	0,0	0,0	0,0	10
RW070	91,6	89,2	88,0	88,3	2,4	3,6	3,3	0
RW102	91,3	91,3	83,7	90,5	0,0	7,7	0,9	8
RW201	86,3	86,3	86,3	86,3	0,0	0,0	0,0	10
RW411	82,1	82,1	82,1	82,1	0,0	0,0	0,0	10
RW316	87,3	87,3	87,3	87,3	0,0	0,0	0,0	10
RW913	87,5	87,5	87,5	87,5	0,0	0,0	0,0	10

Deve observar-se que a heurística GRASP proposta para a programação de veículos obteve o número ótimo de veículos em todas as replicações do algoritmo. No que concerne ao AG desenvolvido para a programação de tripulantes, a heurística supracitada obteve a solução ótima em 19 das 20 instâncias analisadas (a solução ótima do problema 70 não foi encontrada pelo AG).

5. CONCLUSÕES

Neste trabalho foram apresentadas novas formulações para a programação integrada de veículos e de tripulações em sistemas de transporte público, baseadas no *Maximum Covering Problem* – MCP. Tanto quanto é do conhecimento dos autores, não existe nenhum outro trabalho que reporte formulações deste tipo para este problema.

Por outro lado, procurando obter-se um procedimento eficiente e flexível para resolver o problema, desenvolveu-se uma abordagem meta-heurística do tipo GRASP, cujo desempenho, nos experimentos realizados, foi muito bom, visto que foi obtido um *gap* médio de 0,6%, para as 40 instâncias analisadas.

Para o caso do VCSP com *changeovers*, nas 20 instâncias analisadas, pode-se constatar que o tempo computacional médio requerido pelo B&B, para a obtenção da solução ótima, é de apenas 13,5 segundos. Deste modo, a nova formulação proposta representa um ganho do ponto de vista prático, constituindo um instrumento interessante no apoio ao planejamento operacional em empresas de transporte público. O AG desenvolvido para o VCSP com *changeovers* apresentou um desempenho satisfatório, visto que foi obtido um *gap* médio de 0,7% nas 20 instâncias analisadas.

REFERÊNCIAS BIBLIOGRÁFICAS

Ball, M.; Bodin, L.; Dial, R. (1983*) A matching based heuristic for scheduling mass transit crews and vehicles*. Transportation Science, vol. 17, p. 4–31.

Beasley, J. E.; Chu, P.C. (1998) *Constraint handling in genetic algorithms: the set partitioning problem*. Journal of Heuristics, vol. 11, p. 323–357.

Ceder, A. (2002) *Urban transit scheduling: framework, review and examples*. Journal of Urban Planning and Development, v. 128, p. 225 – 244.

Daduna, J. R., Paixão, J. M. P. (1995) *Vehicle Scheduling for public mass transit – an Overview*. In: Daduna, J. R.; Branco, I.; Paixão, J. M. P. (Eds.) Computer-Aided Transit Scheduling. Lecture Notes in Economics and Mathematical Systems, Springer, vol. 430, p. 76–90.

Falkner, J. C.; Ryan, D. M. (1992) *EXPRESS: Set partitioning for bus crew scheduling in Christchurch*. In: Desrochers, M.; Rosseau, J. M. (Eds.) Computer-Aided Transit Scheduling: Proceedings of the Fifth International Workshop, Springer, p. 359–378.

Fischetti, M.; Lodi, A.; Martello, S.; Toth, P. (2001) *A polyhedral approach to simplified crew scheduling and vehicle scheduling problems*. Management Science, vol. 47, n. 6, p. 833–850.

Freling, R., Wagelmans, A. P. M., Paixão, J. M. P. (1999) *An overview of models and techniques for integrating*. In: Wilson, N. H. M. (Ed.) Computer-aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems, Springer, v. 471, p. 441–460, Springer.

Freling, R.; Huisman, D.; Wagelmans, A.P.M. (2001) *Applying an integrated approach to vehicle and crew scheduling in practice.* In: Voβ, S.; Daduna, J. R. (Eds.) Computer-aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems, Springer, v. 505, p. 73–90, Springer.

Freling, R., Huisman, D., Wagelmans, A. P. M. (2003) Models and algorithms for integration of vehicle and crew scheduling. Journal of Scheduling, v. 6, p. 63 – 85.

Friberg, C., Haase, K. (1999) *An exact branch and cut algorithm for the vehicle and crew scheduling problem.* In: Wilson, N. H. M. (Ed.) Computer-aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems, Springer, v. 471, p. 63–80, Springer.

Gaffi, A., Nonato, M. (1999) *An integrated approach to ex-urban crew and vehicle scheduling problem*. In: Wilson, N. H. M. (Ed.) Computer-aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems, Springer, v. 471, p. 103–128, Springer.

Groot, S. W.; Huisman, D. (2008) *Vehicle and crew scheduling: solving large real-world instances with an integrated approach*. In: Hickman, M.; Mirchandani; Voβ, S. (Eds.) Computer-aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems, Springer, v. 600, p. 43 – 56, Springer.

Huisman, D. (2004) *Integrated and dynamic vehicle and crew scheduling*. Ph.D. Thesis. Tinbergen Institute, Erasmus University Rotterdam, Rotterdam.

Huisman, D.; Freling, R.; Wagelmans, A.P.M. (2005) *Multiple-depot integrated vehicle and crew scheduling*. Transportation Science, vol. 39, p. 491–5025.

Laurent, B.; Hao, J.K. (2007) *Simultaneous vehicle and driver scheduling: a case study in a limousine rental company*. Computers & Industrial Engineering, vol. 53, p. 542–558.

Mesquita, M.; Paias, A. (2008) *Set partitioning/covering-based approach for the integrated vehicle and crew scheduling problem*. Computers & Operations Research, vol. 35, p. 1562–1575.

Patrikalakis, G.; Xerokostas, D. (1992) *Experimentation with a new decomposition scheme of the urban public transport scheduling*. In: Desrochers, M.; Rosseau, J. M. (Eds.) Computer-Aided Transit Scheduling: Proceedings of the Fifth International Workshop, Springer, p. 407–425.

Prais, M.; Ribeiro, C. C. (2000) *Reactive GRASP: An application to a matrix decomposition problem in TDMA traffic assignment*. INFORMS Journal on Computing, vol. 12, p. 164–176.

Rodrigues, M. K.; Souza, C.C.; Moura, A.V. (2005) *Vehicle and crew scheduling for urban bus lines*. European Journal of Operational Research , vol. 39, p. 491–5025.

Steinzen, I.; Becker, M.; Suhl, L. (2007) *A hybrid evolutionary algorithm for the vehicle and crew scheduling problem in public transit*. In: 2007 IEEE Congress on Evolutionary Computation – CEC 2007, Singapore.

Bruno de Athayde Prata (baprata@ufc.br)
Centro de Tecnologia, Universidade Federal do Ceará.
Bloco 710 Altos, Campus do Pici s/n, CEP: 60.455-760 – Fortaleza, CE, Brasil.
Teresa Galvão Dias (tgalvao@fe.up.pt)
Departamento de Engenharia Industrial e Gestão, Faculdade de Engenharia da Universidade do Porto.
Rua Dr. Roberto Frias, 4200-465, Porto, Portugal.
Jorge Pinho de Sousa (jsousa@fe.up.pt)
Instituto de Engenharia de Sistemas e Computadores do Porto.
Rua Dr. Roberto Frias, 4200-465, Porto, Portugal.