Novas Soluções para o Problema de Escalonamento de Tripulações

Geraldo Regis Mauri
Luiz Antonio Nogueira Lorena
Instituto Nacional de Pesquisas Espaciais
Laboratório Associado de Computação e Matemática Aplicada
{mauri,lorena}@lac.inpe.br

Resumo

Este trabalho descreve uma metodologia híbrida baseada na aplicação do Algoritmo de Treinamento Populacional (ATP) juntamente com programação linear (PL) para a geração de escalas variadas para tripulações de um sistema de transporte coletivo. Estes métodos são aplicados de maneira interativa, onde o ATP, através de informações da relaxação da PL, é responsável pela geração de boas colunas (baixo custo e boa cobertura das tarefas), e a PL pela resolução de um problema de particionamento de conjuntos formado por essas colunas. Os resultados obtidos são comparados com os da metaheurística Simulated Annealing.

Palavras-chave: *Escalonamento de Tripulações, Algoritmo de Treinamento Populacional, Geração de Colunas.*

1. Introdução

O desenvolvimento de modelos capazes de gerar escalas de tripulações que satisfaçam às restrições impostas, e que possuam o menor custo possível, é um problema desde a década de 60. Isso vem acontecendo devido à contínua transformação dos sistemas de transporte, o que exigem cada vez mais uma gerência eficiente dos recursos disponíveis. Prova disso é a existência de congressos especializados sobre o tema nas últimas décadas, como por exemplo, a *International Conference on Computer-Aided Scheduling of Public Transport* (ver [19], [17], [16], [20], [8], [5] e [3]).

Como é sabido que nas empresas de transporte público urbano a mão-de-obra operacional representa um dos maiores custos [2], uma pequena redução neste item pode significar um ganho considerável no custo total, podendo reduzir significativamente o valor das tarifas para o usuário final, justificando assim qualquer trabalho no sentido de minimizar os custos com a mão-de-obra.

Várias metodologias adotadas para resolver o Problema de Escalonamento de Tripulações (PET) são encontradas na literatura. O problema é conhecido por ser *NP-difícil*. Metaheurísticas como *Algoritmos Genéticos*, *Busca Tabu*, *Simulated Annealing*, entre outras, permitem incluir vários tipos de condições de trabalho facilmente (ver [10], [11], [12], [13], [6] e [9]). Técnicas de Geração de Colunas também têm mostrado resultados muito bons (ver [4], [18] e [3]).

Este trabalho apresenta uma nova alternativa para gerar escalas variadas para tripulações. Tais escalas são geradas com números fixos de tripulações (números próximos aos encontrados nos melhores resultados obtidos em [12]). O método evolutivo chamado Algoritmo de Treinamento Populacional (ATP) é utilizado para gerar colunas de alta qualidade para formar um problema de particionamento de conjuntos, que é resolvido através da programação linear (PL).

O restante do artigo está organizado como segue. A seção 2. apresenta a definição e a modelagem do problema. O ATP é detalhado na seção 3., enquanto a seção 4. descreve o processo de geração de colunas e a interação APT/PL. Os resultados computacionais obtidos são apresentados na seção 5., e as conclusões são resumidas na seção 6..

2. Descrição do Problema

O *Problema de Escalonamento de Tripulações - PET*, também conhecido nos países europeus como *Crew Scheduling Problem - CSP* [5] ou *Bus Driver Scheduling Problem - BDSP* [10], consiste na atribuição da tarefa de condução dos veículos aos motoristas e cobradores (tripulações), de tal forma que as viagens das diferentes linhas atendidas pela empresa sejam executadas com o menor custo possível.

No transporte coletivo rodoviário, usualmente o escalonamento de tripulações é feito após a programação dos veículos. Nesta, as viagens são reunidas em *Blocos*, onde um bloco apresenta a seqüência de viagens distribuídas sucessivamente a um ônibus começando e terminando na garagem. Cada bloco mostra também as *Oportunidades de Troca - OT's* (*Relief Opportunities* segundo [20] ou *Relief Points* segundo [3]), que são intervalos de tempo suficiente para haver a troca das tripulações. A partir das viagens dos blocos

dos veículos são formadas as *Tarefas*, a partir das quais são criados os *Turnos*, que por sua vez são atribuídos às tripulações, as quais irão executá-los durante um dia de trabalho.

O processo de formação desses turnos está sujeito a um conjunto de regras que são específicas a uma organização. Estas regras são geralmente derivadas de regras nacionais e locais, podendo ser obrigatórias ou não. Tipicamente, há restrições no tempo total trabalhado, na extensão total do turno (duração entre o início e o fim do turno) etc.

O problema pode ser descrito como o de formação de uma matriz, onde as colunas representam as tripulações, e as linhas as tarefas. Cada elemento $A_{ij} \in \{0,1\}$, sendo $i \in M = \{1..m\}$ e $j \in N = \{1..n\}$, m é o número de tarefas (linhas), e n o número de tripulações (colunas), onde $a_{ij} = 1$ se a tarefa i pertence ao turno da tripulação j, e 0 caso contrário.

Essa matriz é usada para resolver o seguinte problema de particionamento de conjuntos (PPC):

Minimizar:

$$\sum_{j=1}^{n} c_j x_j \tag{1}$$

Sujeito a:

$$\sum_{j=1}^{n} a_{ij} x_j = 1 \qquad i = 1,...,m \tag{2}$$

$$x_j \in \{0,1\}; \qquad j = 1,...,n \tag{3}$$

onde, c_j representa o custo da coluna j e x_j é igual a 1 se a coluna (turno) j pertencer a solução do problema e 0 caso contrário. Esta é uma formulação clássica e constantemente utilizada em diversos trabalhos encontrados na literatura [4]; [18].

Como citado em [21], na construção de uma escala, a determinação da maneira de se medir o custo, ou seja, a sua "qualidade", não é um trabalho óbvio, pois cada empresa tem sua política de trabalho, obedece às leis vigentes nas localidades onde operam, e ainda impõe as suas próprias regras. Porém, de uma forma geral, essa qualidade é determinada através do número total de tripulações usadas, do custo sobre as horas pagas, de alguma combinação dessas duas, ou ainda através de alguma medida subjetiva de qualidade. Seguindo esta idéia, com o intuito de apresentar soluções variadas para o PET, além das restrições (2) e (3), uma nova restrição está sendo inserida no PPC. Essa nova restrição é definida pela expressão (4).

$$\sum_{j=1}^{n} x_j = Z \tag{4}$$

n representando o número de colunas (tripulações) do problema e Z um número pré-determinado (número de colunas que deverão formarão a solução do problema). Essa abordagem foi baseada nos trabalhos [1] e [14].

Os custos atribuídos às colunas, considerados neste trabalho, penalizam horas extras, tempo ocioso, tempos de sobreposição e excesso no tempo trabalhado. Nestes custos, o *Tempo NORMAL de trabalho* refere-se ao tempo "normal" de duração de um turno (sem horas extras), *Tempo MÁXIMO de trabalho* refere-se ao tempo máximo de duração de um turno (com horas extras), *Tempo ocioso* indica o tempo em que uma tripulação não está trabalhando durante seu turno, *Tempo de sobreposição* indica o tempo em que uma tripulação está, teoricamente, realizando duas tarefas ao mesmo tempo, *Horas extras* indica o tempo em que o turno de uma tripulação excede o tempo normal de trabalho, e *Excesso no tempo total de trabalho* refere-se ao tempo em que um turno excede o tempo máximo de trabalho.

Considera-se obrigatório (essencial) evitar excesso no tempo trabalhado e sobreposições, mas o tempo ocioso e as horas extras são toleráveis (não essenciais). E o PPC requer que todos os custos sejam minimizados enquanto todas as tarefas sejam executadas. A partir desses custos, os turnos são avaliados de acordo com a seguinte expressão:

$$c_j = ET_j + TS_j + HE_j + TO_j \tag{5}$$

onde

$$ET_j = \max(0, [HT(t_{ult}) - HI(t_{pri})] - TMT)$$

$$TS_j = \sum_{i=1}^{p-1} \max(0, [HT(t_i) - HI(t_{i+1})])$$

$$HE_j = \max(0, [HT(t_{ult}) - HI(t_{pri})] - TNT)$$

$$TO_j = \max(0, TNT - [HT(t_{ult}) - HI(t_{pri})]) + ... \\ ... + \sum_{i=1}^{p-1} \max(0, [HI(t_{i+1}) - HT(t_i)])$$

onde, c_j é o custo da tripulação j; ET_j é o excesso no tempo máximo de trabalho da tripulação j; TS_j é o tempo de sobreposição da tripulação j; HE_j é o tempo que excede o tempo normal de trabalho da tripulação j (horas extras); TO_j é o tempo ocioso da tripulação j; TMT é o tempo máximo de trabalho (8hs + 2hs); TNT é o tempo normal de trabalho (8hs); p é o número de tarefas do turno da tripulação j; $HT(t_i)$ é o horário de término da tarefa i; $HI(t_i)$ é o horário de início da tarefa i; t_{pri} é a primeira tarefa do turno da tripulação j; t_{ult} é a última tarefa do turno da tripulação j.

3. Algoritmo de Treinamento Populacional

O Algoritmo de Treinamento Populacional - ATP (ver Figura 1) é um tipo de técnica evolutiva empregado inicialmente em [15], e derivada do Algoritmo Genético Construtivo (AGC).

No ATP, os indivíduos são avaliados diretamente em uma base comum, usando um processo duplo de aptidão, chamado *aptidão-fg*. Esse processo é executado através de heurísticas. Um indivíduo é considerado bem adaptado se ele não melhorar considerando a heurística de treinamento utilizada. A adaptação no treinamento da população é, então, usada para guiar a busca para áreas promissoras.

```
1.  inicializar (P,alpha,t);
2.  enquanto (o critério de parada não for satisfeito)
3.     t ← t+1;
4.     selecionar (P,base,percentual base);
5.     selecionar (P,guia,toda a população);
6.     novo ← recombinar(base,guia);
7.     avaliar (novo);
8.     se (rand() < probabilidade de mutação)
9.        mutação (novo,tamanho da vizinhança);
10.    fim-se
11.    avaliar (vizinhança(novo));
12.    calcular (delta(novo));
13.    se (delta(novo) > alpha)
14.       atualizar (população(P,novo));
15.    fim-se
16.    alpha ← alpha + incrementar (Step);
17.    enquanto (existir delta(S) < alpha)
18.       eliminar (S);
19.    fim-enquanto
20. fim-enquanto
```

Figura 1: Algoritmo ATP.

Neste trabalho, a representação de um turno é dada por: $S_k = (a_{1k}, a_{2k}, ..., a_{mk})$, ou, por exemplo, $S_k = (1,0,1,1,...,0)$, significando que as tarefas 1, 3, 4,..., fazem parte do turno da tripulação *k*.

As duas funções usadas no treinamento evolutivo são definidas através de $g(S_k)$ = "qualidade" do turno *k* (usado durante a geração de colunas - ver expressão 8), e *f(Sk)* = Melhor $g(S'_k)|S'_k \in$ Vizinhança(S_k) (a heurística de treinamento é detalhada na Figura 2).

O processo evolutivo é desenvolvido privilegiando os indivíduos de menor diferença $[g(S_k) - f(S_k)]$ e menor $g(S_k)$ (custo), atribuindo a eles os seguintes *ranks*:

$$\delta(S_k) = d \times [g_{\max} - g(S_k)] - [g(S_k) - f(S_k)] \qquad (6)$$

onde g_{max} é um turno aleatório de custo alto e *d* uma percentagem constante de g_{max}. A população é controlada dinamicamente por um parâmetro de evolução, denominado α, que é atualizado por

$$\alpha = \alpha + Step \times PS \times \frac{\delta_{bst} - \delta_{wst}}{RG} \qquad (7)$$

onde *Step* é uma constante que controla a velocidade do processo evolutivo, *PS* é o tamanho da população corrente, $(\delta_{bst} - \delta_{wst})$ é a variação, no momento, entre os *ranks* do melhor e do pior indivíduo, respectivamente, e *RG* é o número de gerações que faltam para terminar o processo.

O parâmetro α é comparado aos *ranks* (ver expressão (6)), e se $\alpha \geq \delta(S_k)$ então o turno S_k é eliminado da população. A população no momento de evolução α é dinâmica em tamanho e pode ser esvaziada durante o processo.

3.1. Operadores do ATP

Para a aplicação do ATP na geração de colunas, é definida como heurística direcionadora, para o cálculo da função *f*, uma busca local simples, onde são avaliados vários (tamanho da vizinhança definido como parâmetro do ATP) indivíduos alternativos em uma vizinhança. Essa busca é mostrada na Figura 2.

```
1.  seja S_k um turno qualquer;
2.  f(S_k) ← g(S_k);
3.  S'_k ← clone (S_k);
4.  por (tamanho_da_vizinhança vezes)
5.     i ← selecionar (uma tarefa de S'_k);
6.     j ← selecionar (uma tarefa qualquer);
7.     remover (a tarefa i de S'_k);
8.     adicionar (a tarefa j a S'_k);
9.     calcular (g(S'_k));
10.    se(g(S'_k) < f(S_k))
11.       f(S_k) ← g(S'_k);
12.    fim-se
13. fim-por
```

Figura 2: Heurística de treinamento.

A mutação também é baseada em uma busca local, implementada de uma maneira bem simplifica, na qual uma tarefa do turno de uma determinada tripulação é selecionada aleatoriamente, e trocada por outra, também selecionada aleatoriamente. Esse processo é repetido enquanto o novo turno for inválido, isto é, não atender as restrições essenciais. Esse processo pode ser visto na Figura 3.

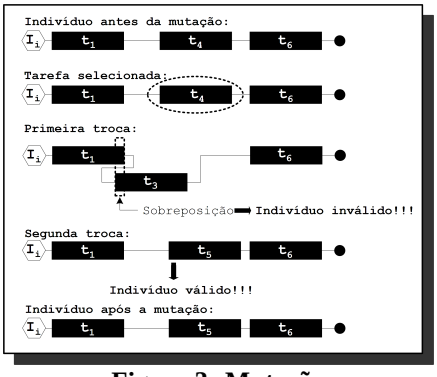

Figura 3: Mutação.

Já o cruzamento, é responsável por gerar novos indivíduos (sempre válidos) da seguinte maneira: dois indivíduos são selecionados (base e guia) e seus cromossomos são fundidos. Como a população é ordenada de maneira decrescente através dos valores dos *ranks* (como na expressão (6)) dos indivíduos, um indivíduo base é selecionado da primeira parte da população, enquanto o guia será selecionado de

toda ela. Feito isso, para garantir que o indivíduo filho seja válido, uma posição do cromossomo fundido é selecionada (aleatoriamente) e as tarefas posicionadas antes dessa posição são removidas, e as tarefas subseqüentes são inseridas de uma forma seqüencial enquanto o novo indivíduo representar um turno válido. Esse processo é descrito na Figura 4.

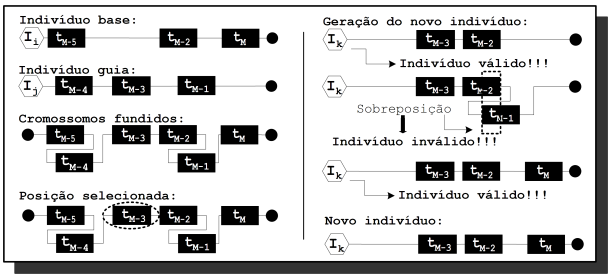

Figura 4: Cruzamento.

É interessante notar que, utilizando esses processos, o ATP formará populações de vários tamanhos, guiados pelo objetivo de selecionar colunas (tripulações) de baixo custo e com uma cobertura suficiente das tarefas. As melhores colunas deverão incluir um número de tarefas diversificado, caracterizado pela heurística de treinamento (função de treinamento) que guiará o processo evolutivo.

4. Interação ATP/PL

Resolver o problema de particionamento de conjuntos (expressões (1)-(4)) pode ser um desafio. Assim, a relaxação de PL é utilizada juntamente com o ATP (ATP/PL) para gerar um conjunto de colunas mais "tratável" por softwares comerciais [7].

Na ATP/PL, uma população inicial é gerada aleatoriamente contendo apenas colunas válidas (colunas que atendam as restrições essenciais) e cobrindo todas as tarefas. A PL é resolvida e novas colunas são geradas através do ATP, considerando os valores das variáveis duais da PL para construir as funções de aptidão f e g. A função $g(S_k)$ é definida por

$$g(S_k) = \frac{c(S_k)}{\sum_{i=1}^{m} \lambda_i a_{ik} + \lambda_{m+1}} \tag{8}$$

$$\theta_k = c(S_k) - \left(\sum_{i=1}^{m} \lambda_i a_{ik} + \lambda_{m+1} \right) \tag{9}$$

onde $c(S_k)$: custo real da coluna k (descrito pela expressão (5)); θ_k: custo reduzido da coluna k; λ_i: variável dual para a tarefa i.

Pode-se observar através das expressões (8) e (9) que, para custos reduzidos negativos, o valor da função g estará no intervalo [0,1]. Então, a heurística de treinamento que define a função f correspondente (melhor g em uma vizinhança), atribuirá uma pequena diferença (g - f) para as colunas que possuírem custos reduzidos negativos. Dessa forma, a população estará então sendo treinada, indiretamente, para geração de indivíduos com custo reduzido negativo (melhorando as colunas para a PL), evitando assim a geração de um número excessivo de colunas, e conseqüentemente, acelerando o processo.

As colunas com custo reduzido negativo ($\theta_k < 0$) são adicionadas ao problema corrente, e o novo PPC é resolvido novamente através da PL. Esse processo é repetido por um certo número de iterações, e quando a solução da PL estabilizar (o mesmo valor objetivo for obtido por, por exemplo, cinco interações consecutivas), ou um certo número de iterações for alcançado, o processo é interrompido.

Entretanto, o valor da solução da PL poderá estabilizar muito cedo, resultando em soluções pobres quando o PPC for resolvido. Então, para evitar que isso aconteça, quando a PL estabilizar, ao invés de adicionar à nova PL apenas as colunas com custo reduzido negativo, todas as colunas geradas pelo ATP são adicionadas. Essa correção renova a geração de colunas e o processo continua convergindo.

Finalmente, terminada essa interação, a PL é convertida em PLI e resolvida pelo software de otimização CPLEX [7]. Porém, o problema final ainda pode ser grande para ser resolvido de uma forma exata, então após algum tempo de execução ou um *gap* limite (porcentagem da diferença dos valores da melhor PLI e PL) a melhor solução encontrada é tomada como solução para o PET. O algoritmo descrito na Figura 5 resume a interação ATP/PL.

```
1.  criar (população direcionada ao problema);
2.  resolver (PL);
3.  enquanto (houver melhora na PL)
4.     executar (ATP);
5.     remover (colunas inválidas);
6.     calcular (custo reduzido das novas colunas);
7.     se (PL não estabilizar)
8.        adicionar (colunas com custo reduzido negativo);
9.     senão
10.       adicionar (todas as colunas geradas);
11.    fim-se
12.    resolver (PL);
13. fim-enquanto;
14. converter (PL para PLI);
15. resolver (PLI);
```

Figura 5: Algoritmo ATP/PL.

5. Resultados Computacionais

Como descrito anteriormente, com o intuito de apresentar soluções variadas para o PET, uma nova restrição (ver expressão 4) foi inserida no PPC. Assim, procurou-se obter soluções com o número de tripulações (fixo) a partir das soluções apresentadas em [12]. Neste trabalho são apresentados os resultados apenas para as instâncias com 25, 50 e 100 tarefas.

Além disso, os resultados são comparados com uma abordagem que utiliza a metaheurística Simulated Annealing (SA) aplicada a um conjunto inicial de colunas geradas aleatoriamente. As penalizações no custo das colunas foram as mesmas definidas na expressão (4). Duas soluções com o SA foram obtidas, sendo que em uma (SA-20), o SA foi executado 20 vezes para diferentes conjuntos iniciais de colunas, e a outra (SA), o SA foi executado apenas uma vez.

Todos os testes foram realizados em um *laptop Toshiba A10 S127* com processador *Intel Celeron*® de 2.0 GHz e 256Mb de memória RAM. Toda a implementação foi desenvolvida na linguagem C++ com chamadas ao software CPLEX (ver [7].

Os resultados obtidos são mostrados nas Tabelas 1, 2 e 3.

Tabela 1: Resultados para 25 tarefas.

25 Tarefas					
Número de tripulações	Método	Tempo ocioso (min.)	Horas extras (min.)	Tempo de sobreposição (min.)	Tempo de execução (seg.)
10	ATP/PL	-	-	-	-
	SA	.	.	.	.
	SA_20	1632	213	38	1070.044
11	ATP/PL	-	-	-	-
	SA	.	.	.	.
	SA_20	1977	98	18	1035.088
12*	ATP/PL	2356	15	0	0.30
	SA	2356	15	0	1.98
	SA_20	2356	15	0	39.60
13	ATP/PL	2821	0	0	0.99
	SA	2821	0	0	0.49
	SA_20	2821	0	0	984.38
14	ATP/PL	3301	0	0	0.80
	SA	3301	0	0	0.46
	SA_20	3301	0	0	999.86

Tabela 2: Resultados para 50 tarefas.

50 Tarefas					
Número de tripulações	Método	Tempo ocioso (min.)	Horas extras (min.)	Tempo de sobreposição (min.)	Tempo de execução (seg.)
18	ATP/PL	-	-	-	-
	SA	.	.	.	.
	SA_20	1813	84	89	1944.26
19	ATP/PL	-	-	-	-
	SA	.	.	.	.
	SA_20	2211	85	6	1898.39
20*	ATP/PL	2600	0	0	41.64
	SA	2607	7	0	95.79
	SA_20	2600	0	0	1915.80
21	ATP/PL	3080	0	0	12.45
	SA	3080	0	0	24.10
	SA_20	3080	0	0	2001.58
22	ATP/PL	3560	0	0	0.37
	SA	3560	0	0	25.19
	SA_20	3560	0	0	1787.73

Nota-se que, em todos os experimentos, as soluções obtidas pelo SA_20 não apresentou excesso no tempo máximo de trabalho, porém nos casos em que os números de tripulações foram inferiores aos obtidos pelas soluções apresentadas em [12] (soluções com o número de tripulações acompanhados do sinal "*") o SA_20 apresentou tempo de sobreposição, ou seja, soluções inválidas.

Tabela 3: Resultados para 100 tarefas.

100 Tarefas					
Número de tripulações	Método	Tempo ocioso (min.)	Horas extras (min.)	Tempo de sobreposição (min.)	Tempo de execução (seg.)
38	ATP/PL	-	-	-	-
	SA	.	.	.	.
	SA_20	6443	0	8	3226.50
39	ATP/PL	-	-	-	-
	SA	.	.	.	.
	SA_20	6917	0	2	3223.65
40*	ATP/PL	7395	0	0	4.98
	SA	7395	0	0	67.19
	SA_20	7395	0	0	1343.80
41	ATP/PL	7875	0	0	2.35
	SA	7875	0	0	52.87
	SA_20	7875	0	0	3302.45
42	ATP/PL	8355	0	0	4.35
	SA	8355	0	0	54.18
	SA_20	8355	0	0	3253.64

Ainda nesses casos, o ATP/PL não obteve soluções, pois tal método não conseguiu gerar um conjunto de colunas (válidas) suficientes para solucionar o PPC de uma forma inteira. Dessa forma, para esses casos o SA não foi utilizado, pois não era necessário comparar o desempenho do ATP/PL. É interessante notar que, o ATP/PL não obteve soluções nesses casos, pois como são inseridas no PPC apenas colunas válidas, o método não foi capaz de encontrar um conjunto de colunas que resolvesse os problemas. Além disso, o SA_20 também não obteve soluções válidas para nenhum desses casos, o que nos leva a acreditar que realmente essas soluções não existem, porém não se pode afirmar isso, pois as soluções exatas para tais casos não são conhecidas. Já para os casos em que o número de tripulações foi maior do que os obtidos pelas soluções apresentadas na primeira abordagem do PET, o ATP/PL assim como o SA e o SA_20 conseguiram obter as mesmas soluções. Para os PT's com 50 e 100 tarefas o desempenho do ATP/PL foi melhor do que o do SA, mostrando assim o potencial desse método, pois mesmo com a inserção de uma nova restrição no problema (ver expressão 4) o método apresentou boas soluções.

6. Conclusões

Este trabalho descreveu uma metodologia híbrida para gerar escalas variadas para tripulações do setor de transporte público. O ATP, integrado com uma técnica tradicional de geração de colunas foi capaz de resolver o subproblema que gera novas colunas de uma forma implícita. Em princípio, a ATP/PL se mostrou satisfatória, pois apresentou melhor um desempenho em relação a metaheurística *Simulated Annea-*

ling.

Além disso, a ATP/PL se mostrou "robusta", pois mesmo com a inserção de uma nova restrição no PPC, o que aumentou significativamente a complexidade do problema, ela apresentou boas soluções.

Enfim, como continuação deste trabalho tem-se o término dos experimentos para os outros PT's utlizados em [12] (250 e 500 tarefas), o que irá completar um conjunto de experimentos cujo fim é a apresentação de soluções diversificadas para o PET.

7. Agradecimentos

Os autores agradecem à Fundação de Amparo à Pesquisa do Estado de São Paulo - FAPESP (processo 03/04547-2) e ao Conselho Nacional de Pesquisas - CNPq (processo 304598/2003-8) pelo apoio financeiro dado ao desenvolvimento deste trabalho.

Referências

[1] Beasley, J. E. e Cao, B. A tree search algorithm for the crew scheduling problem. *European Journal of Operational Research*, 94:517 – 526, 1996.

[2] Bouzada, C. F. 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*, 2002.

[3] Dallaire, A.; Fleurent, C. e Rousseau, J. M. Dynamic Constraint Generation in CrewOpt, a Column Generation Approach for Transit Crew Scheduling. *9th International Conference on Computer-Aided Scheduling of Public Transport (CASPT)*, August 2004.

[4] Desrochers, M. e Soumis, F. A Column Generation Approach to the Urban Transit Crew Scheduling Problem. *Transportation Science*, 23:1 – 13, 1989.

[5] Gintner, V.; Kliewer, N. e Suhl, L. A Crew Scheduling Approach for Public Transit Enhanced with Aspects from Vehicle Scheduling. *9th International Conference on Computer-Aided Scheduling of Public Transport (CASPT)*, August 2004.

[6] Groot, S. e Huisman, D. Vehicle and Crew Scheduling: Solving Large Real-World Instances with an Integrated Approach. *9th International Conference on Computer-Aided Scheduling of Public Transport (CASPT)*, August 2004.

[7] ILOG CPLEX 7.5 Reference Manual. ©Copyright 2001 by ILOG. September 2001.

[8] Kwan, A.; Parker, M.; Kwan, R.; Fores, S.; Proll, L. e Wren, A. Recent Advances in TRACS . *9th International Conference on Computer-Aided Scheduling of Public Transport (CASPT)*, August 2004.

[9] Lee, C. K. The Integrated Scheduling and Rostering Problem of Train Driver Using Genetic Algorithm. *9th International Conference on Computer-Aided Scheduling of Public Transport (CASPT)*, August 2004.

[10] Lourenço, H. R.; Paixão, J. P. e Portugal, R. Multiobjective metaheuristics for the bus-driver scheduling problem. *Transportation Science*, 35:331 – 343, 2001.

[11] Mauri, G. R. Resolução do Problema de Programação de Tripulações de um Sistema de Transporte Público via Simulated Annealing. *Relatório Técnico 02/2003, Departamento de Ciência da Computação - Universidade Federal de Ouro Preto*, 2003. Disponível em: <http://www.lac.inpe.br/~mauri/arquivos/reltec03.pdf>.

[12] Mauri, G. R. e Lorena, L. A. N. Driver Scheduling Generation Using a Population Training Algorithm. *I Brazilian Workshop On Evolutionary Computation. In: SBRN'04 - Brazilian Symposium in Neural Networks*, 2004. Disponível em: <http://www.lac.inpe.br/~mauri/arquivos/mauri_lorena_sbrn04.pdf>.

[13] Mauri, G. R. e Lorena, L. A. N. Método interativo para resolução do problema de escalonamento de tripulações. *XXXVI SBPO - Simpósio Brasileiro de Pesquisa Operacional*, 2004. Disponível em: <http://www.lac.inpe.br/~mauri/arquivos/mauri_lorena_sbpo04.pdf>.

[14] Mingozzi, A.; Boschetti, M. A.; Ricciardelli, S. e Bianco, L. A set partitioning approach to the crew scheduling problem. *Operations Research*, 47(6):873 – 888, Novembro-Dezembro 1999.

[15] Oliveira, A. C. M. Treinamento Populacional em Heurísticas - Aplicações em otimização. *Proposta de tese de Doutorado. Instituto Nacional de Pesquisas Espaciais (INPE)*, 2002.

[16] Shen, Y. e Kwan, R. S. K. Tabu Search for driver scheduling. *Computer-Aided Scheduling of Public Transport - Springer-Velag*, pages 121 – 135, 2001.

[17] Voß, S. e Daduna, J. Computer-Aided Transit Scheduling. *Lecture Notes in Economics and Mathematical Systems. Springer-Verlag*, 505:73 – 90, 2001.

[18] Wilhelm, W. E. A Technical Review of Column Generation in Integer Programming. *Optimization and Engineering*, 2:159 – 200, 2001.

[19] Wilson, N. H. M. Computer-Aided Transit Scheduling. *Proceedings of the Seventh International Workshop on Computer-Aided Scheduling of Public Transport. Springer-Verlag*, 1999.

[20] Wren, A. Scheduling Vehicles and Their Drivers - Forty Years' Experience. *9th International Conference on Computer-Aided Scheduling of Public Transport (CASPT)*, August 2004.

[21] Wren, A.; Fores, S.; Kwan, A.; Kwan, R.; Parker, M. e Proll, L. A flexible system for scheduling drivers. *Journal of Scheduling*, 6:437 – 455, 2003.