MSGVNS: UM ALGORITMO HEURÍSTICO PARA O PROBLEMA DE GERENCIAMENTO DE ESCALA OPERACIONAL DE CONTROLADORES DE TRÁFEGO AÉREO

Bruno Mota Avelar Almeida[1]
Vitor Nazário Coelho[1]
Túlio Ângelo Machado Toffolo[1]
Igor Machado Coelho[2]
Marcone Jamilson Freitas Souza[1]
Bruno Nazário Coelho[1]

[1] Universidade Federal de Ouro Preto / Departamento de Computação / sgt.avelar@gmail.com, vncoelho@gmail.com, tulio@toffolo.com.br, marcone@iceb.ufop.br, brunonazario@gmail.com.
[2] Universidade Federal de Fluminense / Departamento de Computação / igor.machado@gmail.com.

RESUMO
Este artigo apresenta um algoritmo heurístico para resolução de um problema de *timetabling*, o Problema de Gerenciamento de Escalas Operacionais de Controladores de Tráfego Aéreo (PGECTA). O algoritmo proposto combina as metaheurísticas *Multi-Start* e *General Variable Neighborhood Search* (GVNS). Do procedimento *Multi-Start* utilizou-se a fase de construção para produzir soluções viáveis e de boa qualidade rapidamente. O GVNS foi escolhido devido a sua simplicidade, eficiência e capacidade natural de sua busca local, feita pelo método *Variable Neighborhood Descent*, para lidar com diferentes vizinhanças. Resultados computacionais mostram a efetividade do algoritmo proposto.
PALAVRAS-CHAVE: *Timetabling*, Escalonamento dos Controladores de Tráfego Aéreo, Metaheurísticas.

ABSTRACT
This paper presents a heuristic algorithm for solving a timetabling problem, the Problem of Managing Operational Scales of Air Traffic Controllers. (PMSATC). The proposed algorithm combines two metaheuristics, Multi-Start and General Variable Neighborhood Search (GVNS). Multi-Start procedure is used in the construction phase to produce feasible and good solutions quickly. The GVNS was chosen because of its simplicity, efficiency and capacity of its natural local search, made by the Variable Neighborhood Descent method, to deal with different neighborhoods. Computational results show the effectiveness of the proposed algorithm.
KEYWORDS: Timetabling, Air Traffic Controller Rostering, Metaheuristics.

1. INTRODUÇÃO

O presente trabalho trata do Problema de Gerenciamento de Escalas Operacionais de Controladores de Tráfego Aéreo (PGECTA). Neste problema, deve-se determinar uma escala operacional válida para os Controladores de Tráfego Aéreo (*Air Traffic Controller*), atendendo as restrições trabalhistas e operacionais e, se possível, as restrições pessoais.

A maioria dos usuários de transporte aéreo no território brasileiro desconhece a existência de uma grande infraestrutura de segurança envolvendo aeronaves em decolagem, aterrissagem, manobras nos pátios ou estacionadas. O controlador de tráfego aéreo (CTA) é a chave deste grande sistema de gerenciamento e controle do espaço aéreo. O CTA é responsável pela aeronave desde o acionamento do motor até a total parada no aeroporto de destino. Embora de grande importância, a profissão de controlador de tráfego aéreo não é regulamentada no Brasil, além de ser desconhecida do grande público.

O tráfego aéreo brasileiro, bem como o de outros países, tem aumentado consideravelmente nos últimos anos. O CTA deve se adaptar ao volume de tráfego sem perder em qualidade de serviço. Não há espaço para erros humanos, uma vez que um pequeno erro do CTA pode provocar um acidente, como aquele acontecido em 29 de setembro de 2006, quando duas

aeronaves se colidiram no ar, ocasionando a morte de 154 pessoas (http://pt.wikipedia.org/wiki/Voo_Gol_1907). Assim, a profissão exige agilidade intelectual, velocidade de raciocínio, capacidade de adaptação e, principalmente, resistência ao estresse.

O CTA supervisiona uma zona do espaço aéreo, de tamanho variável, denominada setor. O espaço aéreo é cortado por diferentes aerovias, que representam corredores pelos quais as aeronaves se deslocam de um aeroporto a outro. Para todas estas rotas existem regras específicas de navegação, e o CTA deve garantir uma distância mínima entre as aeronaves.

Para uma boa execução de suas atividades, é importante que os CTAs tenham uma escala de trabalho equilibrada ao longo do mês, que respeite todas as normas trabalhistas e que, se possível, atenda às requisições pessoais. Como o PGECTA é uma variante de um problema de *timetabling*, é também um problema da classe NP-difícil (Cabral *et al*., 2000). Ainda não existem algoritmos que o resolvam em tempo polinomial; sendo indicados para sua resolução procedimentos heurísticos.

O presente trabalho apresenta um algoritmo heurístico para resolução do PGECTA. O algoritmo proposto, denominado *MSGVNS*, combina o poderio do procedimento *Multi-Start* – MS (Martí, 2003) com o procedimento GVNS (Hansen *et al*., 2008b,a; Hansen e Mladenovic, 2001; Mladenovic e Hansen, 1997). Do procedimento *Multi-Start* utilizou-se a fase de construção para produzir soluções viáveis e de boa qualidade rapidamente. O GVNS foi escolhido por sua simplicidade, eficiência e capacidade natural de sua busca local, feita pelo método *Variable Neighborhood Descent* – VND, para lidar com diferentes vizinhanças, como relatado em Souza *et al*. (2010). Os algoritmos foram implementados utilizando a versão 1.4 do *framework* OptFrame (Coelho *et al.*, 2011).

O restante deste trabalho está organizado como segue. A Seção 2 detalha o problema abordado. A Seção 3 detalha o algoritmo proposto para resolver o PGECTA. A Seção 4 mostra os resultados dos experimentos computacionais e a Seção 5 conclui o trabalho.

2. PROBLEMA DE GERENCIAMENTO DE ESCALA DOS CONTROLADORES DE TRÁFEGO AÉREO

É comum encontrarmos na literatura trabalhos da área de Pesquisa Operacional relacionados à resolução de problemas como: escalonamento de horários (Mariano, 2007; Freitas, 2007), escalonamento de salas de aulas (Silva *et al*., 2005), escalonamento de motoristas (Méllo *et al*., 2006), escalonamento de enfermeiros (Camilo e Stelle, 2008), entre outros. De modo geral, cada um desses trabalhos propõe métodos para obtenção de soluções boas, quando não ótimas, para o problema abordado. Essas soluções visam ao atendimento de todas as restrições, maximizando o aproveitamento de recursos envolvidos no problema, além de tentar atender ao máximo as preferências das pessoas envolvidas. A diferença entre esses problemas reside, sobretudo, nas regras trabalhistas existentes e em algumas restrições operacionais. O Problema de Gerenciamento de Escala dos Controladores de Tráfego Aéreo (PGECTA) tem algumas características desses problemas apontados, mas difere em outras, como as descritas a seguir.

Cada CTA deve contabilizar no máximo 20 turnos ao final de cada mês, sendo que as restrições operacionais, relacionadas a seguir, não devem ser infringidas:

- Mínimo de 12 CTAs no turno da manhã, 13 no da tarde e 9 no pernoite;

- Máximo de 10 manhãs por mês por CTA;
- Máximo de 10 tardes por mês por CTA;
- Máximo de 10 pernoites por mês por CTA;
- Máximo de 3 pernoites consecutivos por CTA;
- Máximo de 6 serviços consecutivos sem um dia inteiro de folga por CTA;
- Máximo de 6 dias consecutivos de folga por CTA;
- Período mínimo de descanso entre serviços de 1 turno por CTA.

A dificuldade do problema se assenta no fato de, além das restrições operacionais, existirem também requisitos relacionados às preferências de horários por parte dos controladores. Por exemplo, um CTA não pode ou não deseja trabalhar entre os dias 20 e 25 e não poderá trabalhar no turno da manhã entre os dias 5 e 11 do próximo mês. Para este CTA, além das restrições descritas anteriormente, teremos um conjunto de requisitos individuais que poderão ser ou não atendidos.

3. METODOLOGIA

3.1. Representação de uma solução

Uma solução é representada por uma matriz $E_{|T|\times|D|}$, em que:

- $|T| = 3$, são os três turnos possíveis: manhã, tarde, pernoite.
- $|D|$, são o número de dias do mês em que o escalonamento está sendo executado.

A Figura 1 exemplifica uma possível solução para o PGECTA em um mês hipotético de 4 dias. Cada célula e_{ij} na matriz da Figura 1 é um vetor de pertinência de tamanho igual ao número de controladores *nCTA*, isto é, $e_{ij} = [cta_0, cta_1, cta_2, ..., cta_{nCTA}]$

$$s = \left\langle \begin{array}{c|cccc} & Dia1 & Dia2 & Dia3 & Dia4 \\ Manhã & e_{0,1} & e_{0,2} & e_{0,3} & e_{0,4} \\ Tarde & e_{1,1} & e_{1,2} & e_{1,3} & e_{1,4} \\ Pernoite & e_{2,1} & e_{2,2} & e_{2,3} & e_{2,4} \end{array} \right\rangle$$

Figura 1: Representação de uma solução

Cada índice deste vetor pode ter o valor 0 ou 1, sendo que o valor 1 indica que o controlador está trabalhando no turno *i* do dia *j* e o valor 0 indica que o CTA não está trabalhando no turno *i* do dia *j*.

Desconsiderando-se as restrições operacionais, podemos exemplificar parte de uma possível escala dos controladores para três turnos do mês hipotético exposto na solução da Figura 1 por meio dos vetores apresentados na Figura 2.

$$e_{0,1} = [0,1,1,0,1] \qquad e_{1,3} = [1,1,1,1,1] \qquad e_{2,4} = [1,0,0,0,1]$$

Figura 2: Exemplo dos vetores de pertinência

Percebe-se, no primeiro vetor da Figura 2, que os CTAs 2, 3 e 5 estão trabalhando no turno da manhã do dia 1. Para o segundo vetor, notamos que todos os CTAs estão trabalhando no turno da tarde do dia 3. Por último, os CTAs 1 e 5 estão trabalhando no turno da noite do dia 4.

3.2. Estruturas de Vizinhança

Para explorar o espaço de soluções do problema foram propostas duas estruturas de vizinhança, $N^{AR}(.)$ e $N^{T}(.)$, as quais usam dois tipos de movimento a seguir descritos.

3.2.1. Movimento Adiciona e Remove CTA - $N^{AR}(s)$

Este movimento consiste em dar folga a um dado CTA em um turno *i* do dia *j* e colocar um outro CTA para trabalhar no lugar deste. Desta maneira, neste movimento uma célula do vetor e_{ij} muda seu valor de 1 para 0, enquanto uma outra célula deste mesmo vetor tem seu valor alterado de 0 para 1. A Figura 3 apresenta um exemplo de aplicação de um movimento na vizinhança $N^{AR}(s)$, considerando o problema considerado na Figura 1.

$$e_{inicial(2,4)} = [1,0,0,0,1] \qquad e_{final(2,4)} = [0,0,1,0,1]$$

Figura 3: Exemplo de um movimento na vizinhança $N^{AR}(s)$

Na Figura 3, nota-se que o CTA 1 deixou de trabalhar no turno da noite do dia 4, sendo substituído pelo CTA 3.

3.2.2. Movimento Troca CTA - $N^{T}(s)$

Consiste em trocar duas células distintas, ou seja, diferentemente do movimento definido pela vizinhança $N^{AR}(s)$, neste movimento selecionamos um CTA de um vetor e_{ij} e outro CTA de um vetor $e_{n,m}$, levando-se em conta que, se os dias forem iguais ($m = j$), os turnos devem ser diferentes $(n \neq i)$. A Figura 4 apresenta um exemplo de movimento na vizinhança $N^{T}(s)$.

$$e_{inicial(1,2)} = [1,0,0,1,1] \qquad e_{final(1,2)} = [1,0,1,0,1]$$
$$e_{inicial(2,3)} = [1,0,1,0,1] \qquad e_{final(2,3)} = [1,0,0,1,1]$$

Figura 4: Exemplo de um movimento na vizinhança $N^{T}(s)$

Na Figura 4, observa-se que o CTA 4 deixou trabalhar no turno da tarde do dia 2, sendo substituído pelo CTA 3. Além disso, o CTA 4 passou a trabalhar no turno da noite do dia 3, substituindo o CTA 3.

3.3. Avaliação de uma Solução

No modelo heurístico adotado, uma solução *s* é avaliada por uma função *f*, a ser minimizada, dada pela Eq. (1). Esta função é baseada em penalidades, sendo composta de duas parcelas:

$$f(s) = f^{obj}(s) + f^{inv}(s) \tag{1}$$

A parcela $f^{obj}(s)$, dada pela Eq. (2), refere-se à função objetivo do problema, sendo: CTA o conjunto de controladores; *D* o conjunto de dias trabalhados; *T* o conjunto de turnos por dia

trabalhado; $x_{c,t,d}$ uma variável binária que vale 1 se o controlador *c* trabalha no turno *t* do dia *d*, e 0, caso contrário. $P_{c,t,d}$ refere-se ao valor positivo da não preferência do controlador *c* em trabalhar no turno *t* do dia *d*; caso não haja objeção, o valor será 0.

$$f^{obj}(s) = \sum_{c \in CTA} \left(P_{c,t,d} \times x_{c,t,d} \right) \qquad \forall t \in T; d \in D \tag{2}$$

A segunda parcela, $f^{inv}(s)$, dada pela Eq. (3), refere-se às restrições operacionais do problema, descritas na seção 2. Caso alguma dessas restrições seja violada, é atribuído um valor alto à solução s, uma vez que esta solução é inviável na prática.

$$f^{inv}(s) = \sum_{c \in CTA} \left(f_c^{T}(s) + f_c^{T_{M,N,T}}(s) + f_c^{PC}(s) + f_c^{SF}(s) + f_c^{F}(s) + f_c^{MD}(s) \right) \tag{3}$$

Na Eq. (3), a parcela $f_c^{T}(s)$ avalia *s* quanto à inviabilidade de o CTA *c* trabalhar mais que 20 turnos totais, enquanto a parcela $f_c^{T_{M,N,T}}(s)$ avalia se o CTA *c* trabalha mais do que 10 turnos da manhã, da tarde ou da noite; $f_c^{PC}(s)$ avalia *s* quanto ao desrespeito do CTA *c* trabalhar mais do que três pernoites consecutivos; $f_c^{SF}(s)$ avalia *s* quanto ao fato de o CTA *c* trabalhar mais do que seis dias consecutivos sem folga; $f_c^{F}(s)$ avalia *s* quanto à situação de o CTA *c* folgar mais que seis dias consecutivos e $f_c^{MD}(s)$ avalia *s* quanto ao desrespeito ao fato de o CTA c trabalhar dois turnos consecutivos.

3.4. Geração de uma solução inicial

Para geração de uma solução inicial para o problema utiliza-se uma heurística gulosa, cujo pseudocódigo está descrito na Figura 2.

```
   Entrada: Conjunto de dias D a serem alocados
   Saída: Solução de inicial S
1  enquanto D! = ∅ faça
2      Escolha um dia aleatório d ∈ D
3      Remova o dia d do conjunto D
4      t ← manhã
5      Rank_M ← GereRankMelhoresCTAs(t,d)
6      S(t,d) ← Rank_M
7      t ← tarde
8      Rank_T ← GereRankMelhoresCTAs(t,d)
9      S(t,d) ← Rank_T
10     t ← noite
11     Rank_N ← GereRankMelhoresCTAs(t,d)
12     S(t,d) ← Rank_N
13 fim
14 Retorne S
```

Figura 5: Algoritmo ContróiSoluçãoInicial

Na linha 2 da Figura 5, um dia *d* é escolhido aleatoriamente para a alocação dos CTAs. Nas linhas 6, 9 e 12, insere-se na solução inicial *S* do problema os melhores CTAs (escolhidos

pelo procedimento *GereRankMelhoresCTAs*, descrito na Figura 6) para trabalhar no dia *d* nos turnos da manhã, tarde e noite, respectivamente.

Entrada: Turno t, Dia d, Número de CTAs disponíveis $nCTA$
Saída: Vetor v dos *melhores* CTAs para trabalhar no turno t do dia d
1 $Rank \leftarrow$ Inicializa vetor de *RANK* dos CTAs
2 **para** $c \leftarrow 1$ **até** $nCTA$ **faça**
3 $\quad Rank[c] \leftarrow foPenalizacao(c, t, d)$
4 $\quad Rank[c] \leftarrow foDescanso(c, t, d)$
5 $\quad Rank[c] \leftarrow foMaxDiasSemFolga(c, t, d)$
6 $\quad Rank[c] \leftarrow foMaxDiasFolga(c, t, d)$
7 $\quad Rank[c] \leftarrow foMaxTurnoFixoTrab(c, t, d)$
8 $\quad Rank[c] \leftarrow foMaxTurnosTrab(c, t, d)$
9 $\quad Rank[c] \leftarrow foMaxNoitesConsecutivas(c, t, d)$
10 $\quad Rank[c] \leftarrow foMinTurnosTrab(c, t, d)$
11 **fim**
12 $v \leftarrow$ CTAs (necessários para atender o Turno t) com menor valor referente ao vetor $Rank$
13 **Retorne** v

Figura 6: Algoritmo GeraRankMelhoresCTAs

As penalizações das linhas 3, 4, 5, 7, 8 e 9 da Figura 6 são positivas, enquanto as penalizações das linhas 6 e 10 são negativas.

Na linha 3 da Figura 6, o CTA *c* é penalizado caso ele não possa ou não queira trabalhar no turno *t* do dia *d*, sendo o valor dessa penalização dado no problema-teste. Na linha 4, o CTA *c* é penalizado caso ele já esteja alocado para trabalhar no turno anterior ao turno *t* do dia *d*. A linha 5 penaliza o CTA *c* se este trabalhar mais que seis dias seguidos sem folga antes do dia *d*. Na linha 6, o CTA *c* é penalizado caso ele esteja folgando os seis dias anteriores ao dia d. Esta penalização deve ser negativa pois o CTA deve ser alocado o mais rápido possível. A linha 7 penaliza o CTA *c* caso ele já tenha trabalhado 10 turnos no mês da alocação, enquanto a linha 8 penaliza o CTA *c* caso ele esteja trabalhando mais de 20 turnos no mês. A linha 9 penaliza o CTA *c* caso ele esteja trabalhando os últimos três turnos da noite anteriores ao dia *d*. Por último, a linha 10 bonifica (isto é, atribui uma penalização negativa) cada CTA *c* por cada turno não trabalhado. Esta bonificação faz com que os CTAs que estejam trabalhando menos tenham maior chance de serem escolhidos; desta forma, a solução tende a distribuir o número de turnos trabalhados igualmente entre os controladores.

Uma solução inicial para o problema é feita aplicando *MultiStartmax* vezes a fase de construção do procedimento *Multi-Start* (Procedimento *ConstróiSoluçãoInicial* descrito na Figura 5) e retorna a melhor das soluções construídas.

3.5. Algoritmo Proposto

O algoritmo proposto, denominado *MSGVNS*, consiste na combinação dos procedimentos heurísticos *Multi-Start* (Martí, 2003) e GVNS (Hansen *et al*., 2008b,a; Hansen e Mladenovic, 2001; Mladenovic e Hansen, 1997).

O pseudocódigo do algoritmo MSGVNS está esquematizado na Figura 7. Neste algoritmo, *MultiStartmax* representa a quantidade de iterações em que a fase de construção do

procedimento *Multi-Start* é aplicada e *IterMax* indica o número máximo de iterações realizadas em um dado nível de perturbação.

Entrada: Solução s, *MultiStartmax*, *IterMax*, Função $f(.)$
Saída: Solução s^* de qualidade possivelmente superior à s de acordo com a função f

$s_0 \leftarrow s$
$s_0 \leftarrow$ Melhor Solução em $MultiStartmax$ iterações do procedimento *ConstróiSoluçãoInicial*()
$s^* \leftarrow$ VND(s_0, f)
$p \leftarrow 0$
enquanto *critério de parada não satisfeito* **faça**
    $iter \leftarrow 0$
    enquanto $iter < IterMax$ *e critério de parada não satisfeito* **faça**
        $s' \leftarrow$ Refinamento(s^*, p, f)
        se s' *for melhor que* s^* *de acordo com a função* f **então**
            $s^* \leftarrow s'$;
            $p \leftarrow 0$;
            $iter \leftarrow 0$
        fim
        senão
            $iter \leftarrow iter + 1$
        fim
    fim
    $p \leftarrow p + 1$
fim
retorna s^*

Figura 7: Algoritmo MSGVNS

Entrada: r vizinhanças na ordem aleatória: N^{AR}, N^T
Entrada: Solução Inicial s e Funcão de Avaliação f
Saída: Solução s

$k \leftarrow 1$
enquanto $k \leq r$ **faça**
    Encontre o melhor vizinho $s' \in N^{(k)}(s)$
    se $f(s') < f(s)$ **então**
        $s \leftarrow s'$;
        $k \leftarrow 1$
    fim
    senão
        $k \leftarrow k + 1$
    fim
fim
retorna s

Figura 8: Algoritmo VND

A solução inicial s_0 (linha 2 da Figura 7) é gerada aplicando-se *MultiStartmax* vezes a fase de construção do procedimento *Multi-Start*, isto é, o procedimento *ConstróiSoluçãoInicial*, descrito na seção 3.4.

A busca local é feita pelo procedimento VND (vide Figura 8), usando-se os dois movimentos descritos na seção 3.2, ou seja, os movimentos N^{AR} e N^{T}. Para a perturbação, esses dois movimentos são aplicados de forma aleatória de acordo com o procedimento *SelecionaVizinhança* (linha 2 da Figura 9).

Entrada: r vizinhanças: N^{AR}, N^{T}
Entrada: Solução Inicial s, Nível p e Funcão de Avaliação f
Saída: Solução s

1 **para** $i \leftarrow 1$ **até** $p + 2$ **faça**
2 $\quad k \leftarrow$ SelecioneVizinhança(r)
3 $\quad s' \leftarrow$ Perturbação(s, k)
4 **fim**
5 $s \leftarrow \text{VND}(s', f)$
6 **retorna** s

Figura 9: Algoritmo Refinamento

4. RESULTADOS COMPUTACIONAIS

O algoritmo proposto foi implementado em C++ usando o *framework* de otimização *OptFrame* e compilado pelo g++ 4.0. Os testes foram executados em um microcomputador Pentium Core 2 Quad(Q6600), 2.4 GHZ e 8 GB de RAM. Para testá-lo, foi usado um conjunto de duas instâncias, denominadas ***r55_1*** e ***r55_2***, relativas à operação de um centro de controladores de tráfego aéreo brasileiro, considerando dados disponibilizados na Internet. Além destas, foram geradas também duas instâncias adicionais, ***v53_1*** e ***v51_1***. A Tabela 1 mostra as características de cada uma delas.

Tabela 1 : Características das Instâncias

Instância	*nCTA*	*nDias*	*nRestricoesCTA*
r55_1	55	30	43,85
r55_2	55	30	45,72
v53_1	53	30	48
v51_1	51	30	50

Na Tabela 1, a coluna "Instância" indica o nome do problema-teste. A coluna *nCTA* indica o número de controladores disponíveis para alocação e a coluna *nDias* indica o número de dias do mês de alocação. Por fim, a coluna *nRestricoesCTA* indica a média de restrições por CTA, ou seja, o número total de restrições pessoais dividido por *nCTA*.

Após uma bateria de testes preliminares, foram utilizados os parâmetros a seguir. Para a fase de construção da solução inicial: *foDescanso*=5000, *foMaxDiasSemFolga*=5000, *foMaxDiasFolga*=-200, *foMaxTurnoFixoTrab*=5000, *foMaxTurnosTrab*=5000, *foMaxNoitesConsecutivas*=5000 e *foMinTurnosTrab*=-200. Para a função objetivo os pesos de

inviabilidade das parcelas $f_c^T(s)$, $f_c^{T_{M,N,T}}(s)$, $f_c^{PC}(s)$, $f_c^{SF}(s)$, $f_c^F(s)$, $f_c^{MD}(s)$ foram 10000 cada uma. Por último, os parâmetros do *Multi-Start* e do GVNS foram: *MultiStartmax*=300000, *IterMax*=50.

Foram realizadas 100 execuções para cada uma das instâncias com tempo limite de 15s.

A Tabela 2 mostra os resultados obtidos. Nessa tabela, a coluna "Instância" indica o problema-teste considerado. A coluna "Melhor" indica o melhor valor encontrado pelo Algoritmo *MSGVNS* e a coluna "Média" indica a média do *MSGVNS* em 100 execuções.

Tabela 2 : Resultados *MSGVNS*

Instância	Tempo(s)	*MSGVNS*	
		Melhor	Média
r55_1	15	0	4,47
r55_2	15	0	10,85
v53_1	15	176	195,99
v51_1	15	463	501,13

Como podemos observar pela Tabela 2, o algoritmo *MSGVNS* foi capaz de gerar soluções de boa qualidade e baixa variabilidade. Percebe-se que o algoritmo foi capaz de encontrar soluções competitivas em todas as instâncias, encontrando a solução ótima para as instâncias r55_1 e r55_2, já que a penalização total foi nula. Já para as instâncias v53_1 e v51_1, nota-se que o algoritmo *MSGVNS* conseguiu obter soluções factíveis, satisfazendo as restrições pessoais de cerca de 95% dos controladores.

A Tabela 3 mostra as características da melhor solução obtida em cada uma das quatro instâncias utilizadas. Nessa tabela, a coluna *nMaxTurnos* indica o número máximo de turnos que um CTA *c* teve que trabalhar durante o mês enquanto a coluna *nMinTurnos* indica o mínimo de turnos trabalhados. A coluna *nRestricoesVioladas* apresenta o número total de restrições pessoais violadas.

Tabela 3 : Características das melhores soluções

Instância	*nMaxTurnos*	*nMinTurnos*	*nRestricoesVioladas*
r55_1	19	18	0
r55_2	19	18	0
v53_1	20	19	4
v51_1	20	20	10

Analisando-se a Tabela 3, observa-se um equilíbrio em termos de número de turnos trabalhados em todas as quatro instâncias. Nas instâncias r55_1 e r55_2 observa-se que todos os requisitos pessoais dos CTAs foram atendidos, o que equivale ao atendimento de 2412 requisitos na instância r55_1 e 2515 requisitos na instância r55_2. Já para a instância v53_1, apenas 4 dos 2640 requisitos pessoais não foram atendidos, enquanto para a instância v51_1, somente 10 dos 2550 pedidos não foram atendidos.

5. CONCLUSÕES

Este trabalho teve seu foco no Problema de Gerenciamento de Escalas Operacionais dos Controladores de Tráfego Aéreo (PGECTA). Em virtude de sua dificuldade de resolução, foi

proposto um algoritmo heurístico, denominado *MSGVNS*, que combina os procedimentos *Multi-Start* e GVNS.

Utilizando dois problemas-teste envolvendo dados reais e dois gerados de forma aleatória neste trabalho, o algoritmo foi capaz de encontrar soluções de boa qualidade rapidamente e com baixa variabilidade das soluções finais. Além disso, em dois dos problemas-teste, o algoritmo proposto obteve soluções ótimas.

Para trabalhos futuros, propõe-se a concepção de novos tipos de perturbações e ampliação da busca local, realizada pelo procedimento VND. Propõe-se também a implementação do modelo matemático para o problema, buscando, além de uma comparação entre o algoritmo proposto *MSGVNS* e o *solver* utilizado, uma implementação de um algoritmo híbrido, que combinaria a flexibilidade das metaheurísticas com o poderio da programação matemática.

Agradecimentos
Os autores agradecem à Universidade Federal de Ouro Preto, à FAPEMIG e ao CNPq pelo apoio recebido.

REFERÊNCIAS BIBLIOGRÁFICAS

Cabral, L. A. F.; Pontes, R. C. V.; Souza, M. J. F. e Maculan, N. (2000). An heuristic approach for large scale crew scheduling problems at rio sul airlines. *Proceedings of the 40th International Symposium of the AGIFORS*: 1-16.

Camilo, C. e Stelle, D. (2008). Aplicando algoritmos genéticos ao problema de definição de escala de trabalho do corpo de enfermagem de um hospital universitário. *XL Simpósio Brasileiro de Pesquisa Operacional*: 1216-1224, João Pessoa, PB.

Coelho, I. M.; Munhoz, P. L. A.; Haddad, M. N.; Coelho, V. N.; Silva, M. M.; Souza, M. J. F. e Ochi, L. S. (2011). OPTFRAME: A Computational Framework for Combinatorial Optimization Problems. Proceedings of the VII ALIO/EURO Workshop on Applied Combinatorial Optimization: 51-54, Porto.

Freitas, C. C. (2007). Uma ferramenta baseada em algoritmos genéticos para a geração de tabela de horário escolar. *Sétima Escola Regional de Computação*: BAHIA-SERGIPE, Vitória da Conquista/BA.

Hansen, P. e Mladenovic, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research* 130: 449-467.

Hansen, P.; Mladenovic, N. e Pérez, J. A. M. (2008)a. Variable neighborhood search. *European Journal of Operational Research* 191: 593-595.

Hansen, P.; Mladenovic, N. e Pérez, J. A. M. (2008)b. Variable neighborhood search: methods and applications. *4OR: Quarterly journal of the Belgian, French and Italian operations research societies* 6: 319-360.

Mariano, A. S. (2007). Geração automática de grade horária para a faculdade de engenharia elétrica da ufu usando algoritmos genéticos. Mestrado em engenharia elétrica, DEE/UFU, Uberlândia.

Martí, R. (2003). Multi-start methods. Glover, Fred e Kochenberger, Gary, editors, *Handbook of Metaheuristics, volume 57 of International Series in Operations Research & amp; Management Science*: 355-368. Springer New York.

Mladenovic, N. e Hansen, P. (1997). A variable neighborhood search. *Computers and Operations Research* 24: 1097-1100.

Méllo, F. G. A.; Senne, E. L. F. e Lorena, L. A. N. (2006). Uma abordagem para o problema de escalonamento de motoristas. *XXVI ENEGEP*: 1606-1617, Fortaleza, CE.

Silva, A. S. N.; Sampaio, R. M. e Alvarenga, G. B. (2005). Uma aplicação de simulated annealing para o problema de alocação de salas. *Journal of Computer Science – InfoComp* 4: 59-66.

Souza, M. J. F.; Coelho, I. M.; Ribas, S.; Santos, H. G. e Merschmann, L. H. C. (2010). A hybrid heuristic algorithm for the open-pit-mining operational planning problem. *European Journal of Operational Research* 207: 1041-1051.