Proposta de algoritmo genético híbrido para o problema da Árvore Geradora Multiobjetivo baseada no operador *OWA*

Islame Felipe da Costa Fernandes

10 de Outubro de 2017

1 Introdução

Este texto relata a concepção, implementação, experimentos e análises de um algoritmo evolucionário híbrido aplicado ao problema da Árvore Geradora Multiobjetivo com operador *Ordered Weighted Average* (*OWA*), chamada *OWA-ST*. De modo informal, pode-se dizer que o problema da *OWA-ST* consiste em encontrar uma árvore tal que o operador *OWA* seja mínimo. Tal operador, por sua vez, utiliza um vetor de pesos, o qual é aplicado no vetor objetivo ordenado da solução. A *OWA-ST* foi estudada e revisada no primeiro trabalho de DIM0847 - Tópicos Avançados em Algoritmos Experimentais I, onde, na oportunidade, fez-se uma análise experimental dos seus modelos matemáticos (GALAND; SPANJAARD, 2012; FERNÁNDEZ et al., 2017). Assim sendo, omitir-se-á, neste relatório, detalhes da definição matemática e revisão bibliográfica do problema.

A estratégia proposta combina o algoritmo genético clássico com o *simulated annealing* (recozimento simulado). Cria-se, assim, um algoritmo memético inédito para o problema. De fato, até a data de escrita deste texto, a literatura não contemplava nenhuma meta-heurística para o *OWA-ST*. Eis, pois, a principal justificativa deste trabalho.

O objetivo desta pesquisa é analisar o comportamento da abordagem evolucionária híbrida e compará-la com o melhor modelo matemático da literatura (FERNÁNDEZ et al., 2017). Serão relatados experimentos com 10 objetivos, em 60 grafos completos de 30 a 500 vértices, das classes *correlated* e *anticorrelated*. Resultados dos experimentos mostrarão que a meta-heurística oferece, em pouco tempo, uma solução melhor que aquela ofertada pelo *solver* em até uma hora.

Este documento é organizado como segue: Seção 2 detalha o algoritmo proposto e suas principais estratégias, a Seção 3 apresenta os experimentos computacionais e a Seção 4 encerra com as considerações finais e proposta de trabalhos futuros.

2 O algoritmo

O Algoritmo 1 detalha o método proposto, chamado *M-SA*, o qual consiste numa hibridização do Algoritmo Genético (AG) com *Simulated Annealing* (SA). Ou seja, o procedimento pode ser considerado um algoritmo memético, onde a fase de intensificação é feita pelo *SA*.

O primeiro passo do procedimento calcula um conjunto P de soluções iniciais, de tamanho $\#popSize$. A representação das soluções consiste na lista simples das arestas que compõem a árvore, conforme proposto por Raidl e Julstrom (2003). O *fitness* de um indivíduo é igual ao seu valor OWA. A quantidade de gerações é definida pelo parâmetro $\#max\_gen$. Em cada geração, $\#popSize$ indivíduos devem ser gerados e guardados no conjunto Q. Por torneio binário, dois indivíduos P são escolhidos, os quais, com probabilidade $\#propCross$, devem submeter-se ao operador *crossover*. Caso o cruzamento não ocorra, o procedimento *Random Walk* (RAIDL; JULSTROM, 2003) deve gerar um novo indivíduo aleatório que será guardado em Q. Resumidamente, o *Random Walk* constrói uma solução aleatória (com probabilidade uniforme), vértice a vértice, sem considerar os pesos das arestas. Seu tempo esperado é $O(|V|log|V|)$ (RAIDL; JULSTROM, 2003). O objetivo desta estratégia, além de proporcionar diversificação, é garantir que, ao final de uma geração, $\#popSize$ indivíduos tenham sido obtidos.

O novo indivíduo (chamado *filho*) é submetido a um operador de mutação com probabilidade $\#propMutation$. Em seguida, este mesmo indivíduo, seja mutante ou não, é entregue a um procedimento de busca local baseado em recozimento simulado. O objetivo desta fase é intensificar o *fitness* do indivíduo. Porém, como se verá nas subseções subsequentes, o procedimento *SA* almeja, também, garantir um certo grau de diversificação, o que ajuda a fugir de ótimos lacais.

Algoritmo 1: M-SA - Algoritmo genético hibridizado com SA

Entrada: Grafo $G(V, E)$
Saída: Árvore com melhor custo OWA encontrado

```
1  P = getInitialPop(#popSize);
2  best = ∞;
3  setBest(P, best);
4  para g = 1 ... #max_gen faça
5  |  Q = {};
6  |  para r = 1 ... #popSize faça
7  |  |  Selecione randomicamente p1, p2, p3, p4 em P;
8  |  |  pai = torneioBinario(p1, p2);
9  |  |  mae = torneioBinario(p3, p4);
10 |  |  prob = random(0, 1);
11 |  |  se prob<#propCross então
12 |  |  |  filho = crossover(pai, mae);
13 |  |  senão
14 |  |  |  filho = doRandomWalk()
15 |  |  fim
16 |  |  prob = random(0, 1);
17 |  |  se prop<#propMutation então
18 |  |  |  mutation(filho);
19 |  |  fim
20 |  |  SA(filho);
21 |  |  Q = Q ∪ {filho};
22 |  fim
23 |  P = getElite(P ∪ Q);
24 |  setBest(P, best);
25 |  se best não mudar em 5 gerações consecutivas então
26 |  |  renova(P);
27 |  fim
28 fim
29 retorne best
```

Por fim, na linha 23 do Algoritmo 1 são escolhidos os $\#popSize$ indivíduos que irão compor a população da próxima geração. Para tanto, afim de garantir o elitismo na população, ordena-se $P \cup Q$ em ordem crescente de *fitness* e toma-se as $\#popSize$ primeiras soluções. Adota-se ainda, na linha 26, uma estratégia de renovação da população, caso a melhor solução corrente não melhore por cinco gerações consecutivas.

A seguir, detalha-se as principais sub-rotinas da meta-heurística proposta. A Subseção 2.1 descreve como foi gerada a população inicial; a Subseção 2.2 disserta sobre os operadores genéticos implementados; a Subseção 2.3 detalha o procedimento de recozimento simulado; e a Subseção 2.4 disserta sobre a estratégia de renovação da população.

2.1 População inicial

A população inicial é gerada com o auxílio dos métodos *Random Walk* (RAIDL; JULSTROM, 2003) e *rmcPrim*, (KNOWLES, 2002). O primeiro gera uma solução randômica e foi descrito na Seção 2 deste texto. O segundo, por sua vez, inspirado no algoritmo clássico de Prim (1957), constrói uma solução adicionando arestas à solução parcial, as quais são escolhidas entre aquelas pertencentes ao corte formado pelos vértices contidos na solução parcial e os vértices não contidos. Porém, diferente de Prim (1957), a escolha do *rmcPrim* é randômica (probabilidade uniforme) e feita numa lista restritas de arestas candidatas (LRC). A cada iteração, o procedimento toma a aresta e (do corte) que seria escolhida pelo Prim clássico. A lista LRC será composta pelas arestas cujo custo escalarizado seja, no máximo, $\#perTo\%$ maior que e.

O *rmcPrim* deve receber como argumento um vetor de escalarização. Considera-se quatro categorias de vetores de escalarização: as direções de referência do SPEA/R (JIANG; YANG, 2017) e os critérios *k-centrum* (TAMIR, 2001), *k-trimmed* (GALAND; SPANJAARD, 2012) e Hurwicz (HURWICZ, 1951). Sendo M a quantidade de objetivos, a obtenção dos vetores do SPEA/R dar-se pela divisão do espaço objetivo em κ camadas e pelos seguintes passos:

1. Criar um vetor central $\mathbf{C} = (1/M, ..., 1/M)$
2. Criar um conjunto de vetores $\mathbf{B}_i = (b_1, ..., b_M)$ onde $b_i = 1$ e $b_j = 0$ $\forall j \neq i$, $1 \leq j \leq M$, $1 \leq i \leq M$; $\mathbf{B}_i$ é o *i-ésimo* ponto extremo e referente ao eixo do *i-ésimo* objetivo;
3. Criar κ vetores entre $\mathbf{C}$ e $\mathbf{B}_i$. O *r-ésimo* vetor, $r \in \{1, ..., \kappa\}$, $\mathbf{D}_i^r$, é gerado pela seguinte equação: $\mathbf{D}_i^r = \mathbf{C} + \frac{r}{\kappa}(\mathbf{B}_i - \mathbf{C})$;
4. Finalmente, para cada camada $r \in \{1, ..., \kappa\}$ e cada $t \in \{1, ..., r\}$, computar os seguintes pontos dentro da camada $\hat{\mathbf{D}}_{i,r}^t = \mathbf{D}_i^t + \frac{t}{r+1}(\mathbf{D}_{i+1}^r - \mathbf{D}_i^t)$;

O *k-centrum* consiste em sortear, com probabilidade uniforme, um inteiro $k \in [1, M]$ e gerar os pesos tais que $\omega_1 = ... = \omega_k = 1/k$ e $\omega_{k+1} = ... = \omega_M = 0$. O *k-trimmed* sorteia um inteiro $k \in \left[1, \frac{M}{2}\right[$ e atribui $\omega_1 = ... = \omega_k = 0$, $\omega_{k+1} = ... = \omega_{M-k} = \frac{1}{M-2k}$ e $\omega_{M-k} = ... = \omega_M = 0$. Por fim, o critério *Hurwicz* determina um α e atribui $\omega_1 = \alpha$, $\omega_2 = ... = \omega_{M-1} = 0$ e $\omega_M = 1 - \alpha$. A presente implementação sorteia, com probabilidade uniforme, $\alpha \in \{0.2, 0.3, 0.4, 0.6, 0.7, 0.8\}$. Esta variedade de vetores de escalarização proporciona a criação de soluções bem diversificadas, isto é, bem espalhadas pelo espaço de busca.

O procedimento de criar soluções iniciais executa $\#popSize$ iterações. Em cada iteração, gera-se, inicialmente, uma solução randômica com o *Random Walk*. Em seguida, sorteia-se um vetor de cada categoria e aplicada-se o *rmcPrim* a cada vetor, gerando quatro soluções diferentes. Deve ser inserida na população aquela que possuir menor *fitness*.

2.2 Operadores genéticos

O algoritmo proposto utiliza os operadores genéticos clássicos: *crossover* e mutação. Para o caso específico da árvore geradora, Raidl e Julstrom (2003) propuseram procedimentos eficientes para tais operadores, além de também propor diretrizes gerais para algoritmos evolucionários que trabalham com problemas de árvores.

Sejam T_1 e T_2 dois indivíduos (árvores) quaisquer. O *crossover* consiste em obter uma árvore geradora a partir do grafo formado pelas arestas de $T_1 \cup T_2$. Com isso, visa-se obter uma árvore geradora que possua arestas de ambos os progenitores (T_1 e T_2). Raidl e Julstrom (2003) afirmam que qualquer procedimento para obter uma árvore randômica pode ser utilizado. O procedimento *Random Walk* foi adotado. Raidl e Julstrom (2003) afirmam ainda que estratégia de cruzamento pode falhar em problemas de árvores que possua restrições adicionais (o que exigiria a adição de arestas aleatórias advindas de $E-(T_1 \cup T_2)$). Este problema não ocorre no caso da *OWA-ST*.

O operador de mutação deve reintroduzir informação genética no indivíduo. Assim, sendo T_1 uma solução, tal operador foi implementado em $O(|V|)$ de tal sorte que duas arestas são removidas de T_1 e seus vértices são reconectados escolhendo duas arestas viáveis em $E-T_1$. A escolha desta estratégia fundamenta-se na necessidade de garantir que uma solução mutante, uma vez deslocada de sua região original no espaço objetivo, possa ter a chance de ser intensificada pelo procedimento de recozimento simulado e, posteriormente, integrar a nova população.

2.3 *Simulated Annealing*

O procedimento geral do recozimento simulado pode ser apreciado no Algoritmo 2. O objetivo do algoritmo é intensificar o *fitness* da solução passada como argumento, garantindo, simultaneamente, um certo grau de diversificação. Eis, pois, a justificativa de adoção da referida meta-heurística: um razoável balanceamento dentre explotação e exploração num único procedimento de busca.

Os parâmetros T_0 e L_0 denotam, respectivamente, a temperatura inicial e a quantidade de iterações nesta temperatura. Há ainda os parâmetros gerais F_T e F_L, que denotam, respectivamente, o fator de decaimento e o fator de crescimento da quantidade de iterações por temperatura. Ou seja, cai a temperatura, cresce a quantidade de iterações.

A cada iteração, uma solução vizinha da solução corrente s é determinada randomicamente. O algoritmo sorteia uniformemente duas arestas, remove-as de s e religa seus vértices como no *2-OPT*. Tal estratégia de vizinhança foi utilizado por Rocha, Goldbarg e Goldbarg (2006) em sua busca tabu. Se o *fitness* da solução s' é melhor que o de s, então aquela passa a ser a solução corrente. Caso contrário, s' se torna a solução corrente com probabilidade $exp\{-\frac{\Delta C}{T_i}\}$. Deste modo, perceba que quanto maior a temperatura, maior a probabilidade de s' ser considerada nas próximas iterações, mesmo que seu *fitness* não seja tão bom. Isso garante a diversificação. Conforme a temperatura diminui, tal

probabilidade tende a zero e, assim, a diversificação tende a dar lugar à intensificação.

Algoritmo 2: *Simulated Annealing*

Entrada: sol: solução, T_0: temperatura inicial, L_0: quantidade inicial de iterações

```
1  s = sol;
2  T_i = T_0;
3  L_i = L_0;
4  enquanto T_i > 0 faça
5      para i=1 ... L_i faça
6          escolha randomicamente dois pares de aresta (v_l, v_{l+1}) e (v_j, v_{j+1}) em s;
7          remova (v_l, v_{l+1}) e (v_j, v_{j+1}) de s;
8          religue os vértices v_l, v_{l+1}, v_j, v_{j+1} e produza s';
9          ΔC = OWA(s') − OWA(s);
10         se ΔC ≤ 0 então
11             s = s';
12             se OWA(s) < OWA(sol) então
13                 sol = s;
14             fim
15         senão
16             se random(0,1) < exp{−ΔC/T_i} então
17                 s = s'
18             fim
19         fim
20     fim
21     T_i = T_i / F_T;
22     L_i = L_i * F_L;
23 fim
```

2.4 Estratégia de renovação

O algoritmo proposto adota ainda uma estratégia de renovação da população. Inspirando-se no trabalho de Rocha, Goldbarg e Goldbarg (2006), a população deve ser renovada sempre que a melhor solução corrente não mudar em cinco iterações consecutivas.

O procedimento de renovação adotado se divide em duas partes: na primeira, 50% da população é substituída por um indivíduo gerado randomicamente pelo procedimento *rmcPrim*, onde o vetor de escolarização é o vetor w (pesos OWA passado pela instância). Na segunda parte, um terço da população deve sofrer mutação (conforme o operador descrito na subseção 2.2). Em ambas as partes, os indivíduos a serem substituídos são escolhidos aleatoriamente com probabilidade uniforme.

O objetivo desta estratégia de renovação é proporcionar a diversificação da população, criando novos indivíduos que poderão ser capazes de melhorar o ótimo corrente.

3 Experimentos computacionais

Apresenta-se, nesta seção, os resultados dos experimentos computacionais do algoritmo proposto. Todos os experimentos foram realizados numa máquina Intel *Xeon W3520*, 2.67 GHz, Sistema Operacional Ubuntu 14.04 *LTS*, 64 bits, 8GB de memória RAM. Os algoritmos foram implementados em linguagem *C++* e compilados com compilador *GNU g++* versão 4.8.4. O modelo de programação matemática de Fernández et al. (2017) foi implementado com o *solver Gurobi* versão 6.5.

3.1 Instâncias utilizadas

Os experimentos foram efetuados em grafos completos, com 10 objetivos, tendo de 30 a 500 vértices. Para cada quantidade n de vértices, foram geradas três instâncias distintas, denotadas, respectivamente, por $n.1$, $n.2$ e $n.3$. Todas as instâncias foram geradas com o gerador de Knowles (2002) considerando a correção proposta por Chen et al. (2007) (que impede de gerar arestas com peso negativo). Knowles (2002) classificou suas instâncias em três subgrupos: *correlated*, *anti-correlated* e *concave*. As duas primeiras dizem respeito à correlação β dos pesos das arestas (positivo e negativo, respectivamente). Os valores escolhidos para β foram 0.2, 0.5 e 0.85 para instâncias *correlated n.1*, *n.2* e *n.3*, respectivamente. Para instâncias *anti-correlated*, o parâmetro β assume valores -0.2, -0.5 e -0.85 respectivamente para os grupos *n.1*, *n.2* e *n.3*. Até a data dos experimentos aqui relatados, o gerador de Knowles (2002) não contemplava instâncias *concave* com mais de dois objetivos. Tal classe de instâncias foi, por isso, desconsiderada.

As instâncias precisam ser acrescidas de um vetor $w \in \Re^M$ de pesos. Os critérios conhecidos para gerar o vetor w são *k-centrum* (TAMIR, 2001), *k-trimmed* (GALAND; SPANJAARD, 2012) e o critério de Hurwicz (HURWICZ, 1951). O segundo foi adotado pelo presente trabalho. Para definir o vetor w, tal critério sorteia um inteiro $k \in \left[1, \frac{M}{2}\right[$ e atribui $w_1 = ... = w_k = 0$, $w_{k+1} = ... = w_{M-k} = \frac{1}{M-2k}$ e $w_{M-k} = ... = w_M = 0$. O valor de k adotado é sorteado com probabilidade uniforme em $\{2, 3, 4\}$.

3.2 Parâmetros do *M-SA*

Os valores dos parâmetros do *M-SA* foram fornecidos a partir de experimentos preliminares realizados por esta pesquisa, buscando sempre conciliar qualidade de solução e tempo de processamento. A Tabela 1 expõe os valores sugeridos.

Tabela 1: Parâmetros para o *M-SA*

$\#max\_gen = 50$	$T\_0 = 30$
$\#popSize = 100$	$F\_T = 1,8$
$\#propCross = 0,97$	$L\_0 = 15$
$\#propMutation = 0,1$	$F\_L = 1,8$
$\#perTo = 0,03$	$\kappa = 6$

3.3 Metodologia dos experimentos

Para cada instância, o *M-SA* foi executado 30 vezes. Foram observados dados como qualidade final da solução retornada, evolução da qualidade ao longo das gerações do processo evolucionário, tempo para atingir a melhor solução, tempo total, quantidade de vezes em que a população precisou ser renovada e a contribuição dos operadores de *crossover*, mutação e do *SA*. Analisar a contribuição dos referidos operadores significa analisar a quantidade de vezes em que eles conseguiram, de fato, melhorar o *fitness* de uma solução. Falar-se-á, pois, em taxa de sucesso destes operadores.

O *M-SA* será comparado com o melhor algoritmo exato da literatura para o *OWA-ST*. Fernández et al. (2017) propuseram um modelo matemático para o problema, o qual, segundo os autores e segundo experimentos preliminares realizados por esta pesquisa, apresenta excelentes resultados, principalmente para instâncias de grande porte (por exemplo, com 10 objetivos e mais 40 vértices). Fernández et al. (2017) ainda propuseram melhorias em seu modelo, diminuindo o conjunto de restrições e variáveis. Segundo os autores, tal modelo melhorado chega a ser melhor que aquele proposto por Galand e Spanjaard (2012).

Esta pesquisa comparou, em termos de qualidade de solução, o modelo melhorado de Fernández et al. (2017) com o *M-SA*. A execução do modelo foi limitada a uma hora. Assim, visa-se verificar o quão boa é a qualidade da solução retornada pelo *M-SA* quando comparada com aquela retornada pelo *solver* em até uma hora. Com este propósito, calculou-se o desvio percentual segundo a fórmula:

$$d_p = \frac{OWA(s_h) - OWA(s_s)}{OWA(s_s)} * 100 \tag{1}$$

onde s_s e s_h são, respectivamente, a solução retornada pelo *solver* e pelo *M-SA*. O desvio percentual expressa o quão distante s_h está de s_s, com relação ao custo OWA. Note que, como o *solver* foi limitado a uma hora, então pode acontecer de $OWA(s_h) < OWA(s_s)$, o que implica $d_p < 0$, indicando que a solução heurística conseguiu ser melhor que a do solver limitado.

Finalmente, esta pesquisa calculou o desvio percentual para as soluções de custo médio, mediano e mínimo das 30 execuções do *M-SA* para cada instância. O objetivo é analisar como se comporta o algoritmo nestes três aspectos, verificando, pois, se existe diferença entre eles.

3.4 Resultados

A Tabela 2 compara a qualidade média, mediana e mínima de 30 execuções do *M-SA* com a qualidade da solução do *solver* até uma hora. A referida tabela também mostra o tempo total (em segundos) da meta-heurística e o tempo (em segundos) que a mesma necessitou para alcançar a melhor solução. Os resultados são divididos em *correlated* e *anticorrelated*.

Em todas as instâncias, de tamanho até 400 vértices, de ambas as classes, o *solver* atingiu o tempo limite mas conseguiu retornar uma solução factível. Para as instâncias de tamanho 500, entretanto, o *solver* não conseguiu retornar uma solução em até uma hora. Deste modo, o símbolo — utilizado na Tabela 2 remete a este último fato.

Em todas as instâncias *correlated*, o *M-SA* conseguiu solações com custo OWA menor que aquela retornada pelo *solver* ao fim de uma hora. Tal fato pode ser constatado tanto observando diretamente os custos OWA expostos na tabela quanto observando que os desvios percentual estão negativos. Pela fórmula (1), o valor negativo do desvio percentual significa que o custo da solução heurística foi melhor (ultrapassou o *solver*). Mais que isso, o *M-SA* conseguiu este feito para os custos médio, mediano e mínimo das 30 execuções. A única exceção é para a instância *correlated* 45.3, onde a solução mediana apresentou custo $0,14\%$ superior ao custo da solução *solver*. Este caso foi isolado na classe *correlated*. De modo geral, a vantagem da meta-heurística, em termos de qualidade da solução, em relação ao *solver*, cresce com o tamanho da instância, ficando bastante evidente para instâncias com mais de 100 vértices, conforme se observa a partir da tendência decrescente dos desvios percentual na Tabela 2. Além disso, o *M-SA*, como era esperado, consumiu, em média, um tempo total consideravelmente inferior, não ultrapassando os 60 segundos.

O comportamento do *M-SA* nas instâncias *anticorrelated* é análogo ao comportamento nas *correlated*. Em apenas uma instância *anticorrelated*, a saber 40.2, o *M-SA* obteve soluções de custo médio $1,25\%$ superior a do *solver*. Neste mesmo caso, as soluções de custo mediano e mínimo (das 30 execuções) apresentam valor OWA, respectivamente, $1,41\%$ e $0,39\%$ superior ao valor OWA da solução do *solver*. Este caso também é isolado. Em todas as outras instâncias, o *M-SA* obteve os melhores custos, conforme se observa nos valores negativos do desvio percentual. Novamente, quanto maior o tamanho da instância, maior a distância entre o custo heurístico e o do *solver*, o que aumenta a vantagem da meta-heurística.

Tabela 2: Comparação do *M-SA* com o resultado do solver limitado em instâncias *k-trimmed* com 10 objetivos

Instância	Correlated										Anticorrelated									
	Solver		M-SA								Solver		M-SA							
	Melhor valor OWA	Tempo(s)	Solução média	Desvio Percentual da solução média	Solução mediana	Desvio Percentual da solução mediana	Solução mínima	Desvio Percentual da solução mínima	Tempo(s) médio para encontrar a solução mínima	Tempo(s) médio total	Melhor valor OWA	Tempo(s)	Solução média	Desvio Percentual da solução média	Solução mediana	Desvio Percentual da solução mediana	Solução mínima	Desvio Percentual da solução mínima	Tempo(s) médio para encontrar a solução mínima	Tempo(s) médio total
30.1	880,75	3600,00	856,61	-2,74	860,50	-2,30	814,75	-7,49	1,38	2,83	914,00	3600,00	834,00	-8,75	836,50	-8,48	812,50	-11,11	0,87	2,82
30.2	662,50	3600,00	653,57	-1,35	653,25	-1,40	647,00	-2,34	0,57	2,80	707,34	3600,00	685,04	-3,15	685,17	-3,13	683,50	-3,37	0,22	2,80
30.3	283,00	3600,00	280,05	-1,04	280,00	-1,06	279,75	-1,15	0,02	2,81	299,25	3600,00	295,13	-1,38	295,00	-1,42	294,25	-1,67	0,65	2,74
35.1	873,50	3600,00	842,05	-3,60	841,00	-3,72	831,00	-4,87	0,52	3,11	1056,00	3600,00	1 024,75	-2,96	1 024,25	-3,01	1017,00	-3,69	0,99	3,01
35.2	760,00	3600,00	741,02	-2,50	741,42	-2,45	732,17	-3,66	0,74	3,08	842,84	3600,00	814,92	-3,31	815,17	-3,28	811,00	-3,78	0,43	3,08
35.3	339,00	3600,00	337,27	-0,51	337,25	-0,52	335,25	-1,11	0,35	3,12	306,25	3600,00	306,03	-0,07	306,00	-0,08	306,00	-0,08	0,16	3,05
40.1	1 196,17	3600,00	1 129,52	-5,57	1 131,09	-5,44	1 116,17	-6,69	1,72	3,39	1 295,84	3600,00	1 219,66	-5,88	1 219,76	-5,87	1 215,17	-6,23	0,72	3,41
40.2	855,25	3600,00	837,39	-2,09	837,50	-2,08	836,00	-2,25	0,08	3,35	831,50	3600,00	841,93	1,25	843,25	1,41	834,75	0,39	1,22	3,39
40.3	363,00	3600,00	354,92	-2,23	354,50	-2,34	353,00	-2,75	0,40	3,36	330,00	3600,00	328,80	-0,36	329,00	-0,30	326,00	-1,21	0,27	3,40
45.1	1 284,17	3600,00	1 241,74	-3,30	1 245,17	-3,04	1 222,34	-4,81	0,80	3,82	1410,00	3600,00	1 316,30	-6,65	1 318,42	-6,50	1299,00	-7,87	1,32	3,79
45.2	965,00	3600,00	942,93	-2,29	943,50	-2,23	933,50	-3,26	0,79	3,85	990,25	3600,00	948,83	-4,18	947,50	-4,32	939,00	-5,18	1,73	3,73
45.3	361,50	3600,00	360,98	-0,14	362,00	0,14	354,50	-1,94	0,35	3,78	394,17	3600,00	388,98	-1,32	388,83	-1,35	388,83	-1,35	0,10	3,77
50.1	1 451,25	3600,00	1 371,53	-5,49	1 370,50	-5,56	1 365,50	-5,91	1,22	4,07	1452,00	3600,00	1 377,19	-5,15	1 375,50	-5,27	1 359,75	-6,35	2,16	4,13
50.2	996,00	3600,00	962,87	-3,33	965,00	-3,11	941,00	-5,52	1,54	4,09	1091,00	3600,00	1 043,47	-4,36	1 046,50	-4,08	1 024,50	-6,10	1,26	4,12
50.3	407,50	3600,00	402,23	-1,29	402,42	-1,25	401,00	-1,60	0,78	4,12	434,00	3600,00	432,59	-0,33	432,50	-0,35	430,67	-0,77	0,54	4,11
100.1	3 357,25	3600,00	2 515,76	-25,06	2 517,38	-25,02	2 491,50	-25,79	6,01	9,14	3752,00	3600,00	2 447,88	-34,76	2 446,25	-34,80	2 424,50	-35,38	3,06	9,29
100.2	2 597,84	3600,00	1 860,22	-28,39	1 860,34	-28,39	1 851,50	-28,73	2,04	9,24	2 901,67	3600,00	1 947,35	-32,89	1 947,67	-32,88	1 944,67	-32,98	0,81	9,20
100.3	994,50	3600,00	717,30	-27,87	717,88	-27,82	711,00	-28,51	2,23	9,25	962,00	3600,00	730,12	-24,10	730,50	-24,06	726,00	-24,53	1,72	9,09
200.1	7 207,51	3600,00	4 745,11	-34,16	4 744,60	-34,17	4 737,51	-34,27	8,28	23,31	7011,00	3600,00	4 369,60	-37,68	4369,00	-37,68	4330,00	-38,24	13,98	23,18
200.2	4541,00	3600,00	3 104,35	-31,64	3 106,25	-31,60	3072,00	-32,35	13,67	23,24	6307,00	3600,00	3 543,57	-43,82	3 542,75	-43,83	3 522,50	-44,15	11,67	22,36
200.3	1 445,75	3600,00	1 270,24	-12,14	1 270,50	-12,12	1 263,25	-12,62	5,78	23,51	2 516,84	3600,00	1 311,32	-47,90	1 311,26	-47,90	1 309,17	-47,98	4,69	23,29
300.1	11802,00	3600,00	5 930,23	-49,75	5 925,25	-49,79	5894,00	-50,06	17,33	29,05	13001,00	3600,00	6 189,25	-52,39	6190,00	-52,39	6161,00	-52,61	14,03	29,29
300.2	12194,00	3600,00	4 286,07	-64,85	4284,00	-64,87	4 271,50	-64,97	15,90	29,03	17376,00	3600,00	4 632,45	-73,34	4 629,25	-73,36	4 607,50	-73,48	17,96	29,26
300.3	7141,00	3600,00	1 686,50	-76,38	1 686,50	-76,38	1679,00	-76,49	11,19	29,14	7921,00	3600,00	1 809,20	-77,16	1 809,12	-77,16	1 804,25	-77,22	8,74	29,26
400.1	17258,00	3600,00	7 679,32	-55,50	7 680,25	-55,50	7656,00	-55,64	22,76	40,36	17899,00	3600,00	7 896,62	-55,88	7 900,75	-55,86	7846,00	-56,17	27,87	40,24
400.2	11480,00	3600,00	6 230,93	-45,72	6 231,68	-45,72	6 219,35	-45,82	3,88	40,72	18 864,20	3600,00	6 363,14	-66,27	6 362,38	-66,27	6 336,50	-66,41	29,57	40,51
400.3	11261,00	3600,00	2 249,72	-80,02	2 249,88	-80,02	2 244,25	-80,07	3,06	41,07	10947,00	3600,00	2 257,52	-79,38	2 258,50	-79,37	2 241,50	-79,52	25,62	40,36
500.1	—	—	10371,00	—	10 370,10	—	10 353,40	—	19,82	52,30	—	—	10 823,60	—	10827,00	—	10 766,20	—	37,17	51,98
500.2	—	—	7 234,22	—	7234,00	—	7211,00	—	28,91	52,07	—	—	7 725,93	—	7 723,50	—	7702,00	—	41,69	51,63
500.3	—	—	2 606,28	—	2606,00	—	2 603,25	—	11,27	52,78	—	—	2 752,22	—	2 752,62	—	2 736,25	—	19,56	52,81

As Figuras 1 e 2 mostram, graficamente, o tempo médio total, em segundos, consumido pelo *M-SA* e o tempo médio para encontrar a melhor solução, respectivamente, para instâncias *correlated* e *anticorrelated*. Ambos os tempos começam a ser contabilizados imediatamente antes da primeira geração. A proximidade entre estes tempos sugere que o *M-SA* conseguiu melhorar o ótimo corrente nas últimas gerações. O contrário significa que o algoritmo não conseguiu evoluir significativamente o ótimo corrente, o qual, provavelmente, pode configurar-se como ótimo local. No geral, à luz das Figuras 1 e 2, o *M-SA* foi mais eficaz em evoluir (melhorar) as instâncias *anticorrelated*. Na classe *correlated*, a evolução estancou mais frequentemente em ótimos locais. Para ilustrar tal fenômeno, aconselha-se observar as Figuras de 3 a 6. Nelas, plota-se a curva de evolução do ótimo corrente em uma execução, respectivamente, para as instâncias 500.2 *correlated*, 500.2 *anticorrelated*, 300.1 *correlated* e 300.1 *anticorrelated*. Neste exemplo, o algoritmo consegue, na classe *anticorrelated*, levar a melhoria da solução corrente até às últimas iterações. Na classe *correlated*, porém, a melhoria estaciona bem antes da última geração.

Figura 1: Tempo (s) médio total e tempo (s) médio que o *M-SA* levou para atingir a melhor solução para instâncias correlated

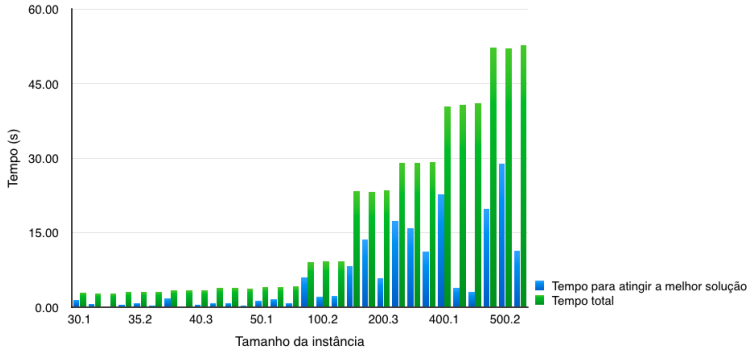


Figura 2: Tempo (s) médio total e tempo (s) médio que o *M-SA* levou para atingir a melhor solução para instâncias anticorrelated

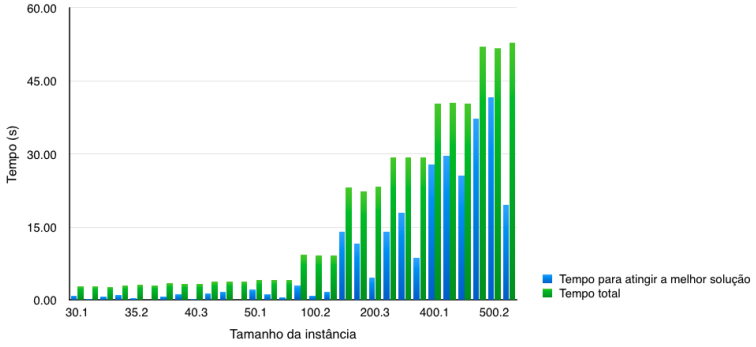


Figura 3: Evolução da solução de menor custo para a instância 500.2 corrlated

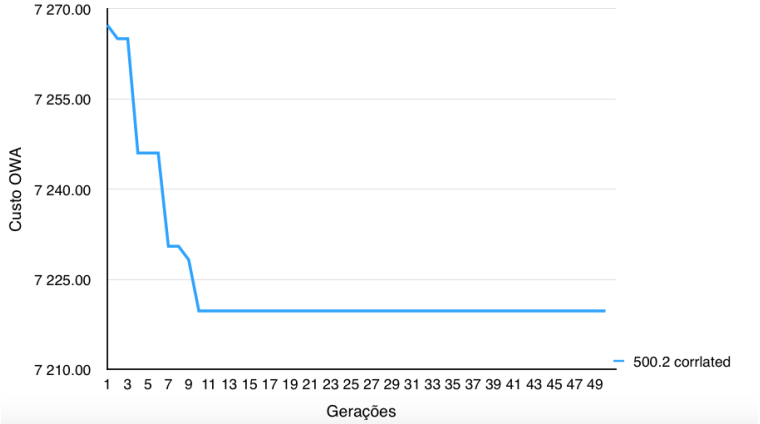


Figura 4: Evolução da solução de menor custo para a instância 500.2 anticorrlated

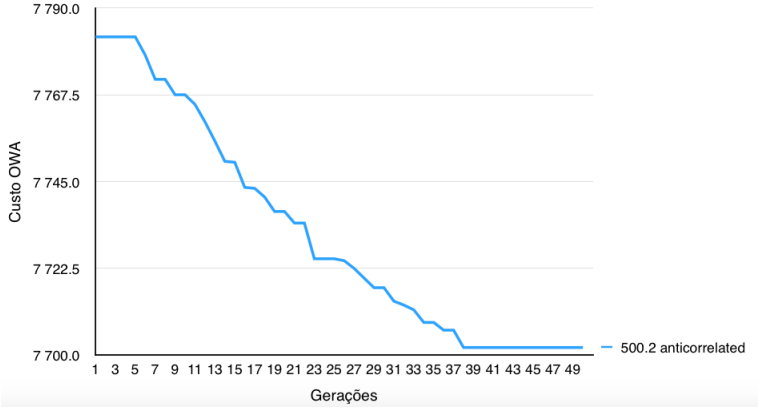


Figura 5: Evolução da solução de menor custo para a instância 300.1 corrlated

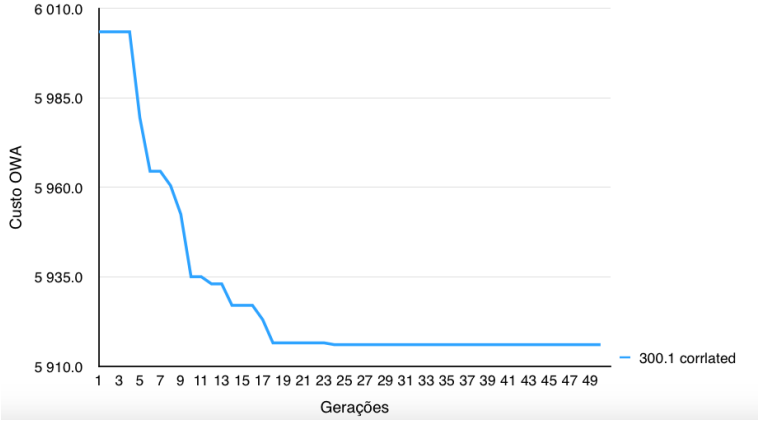
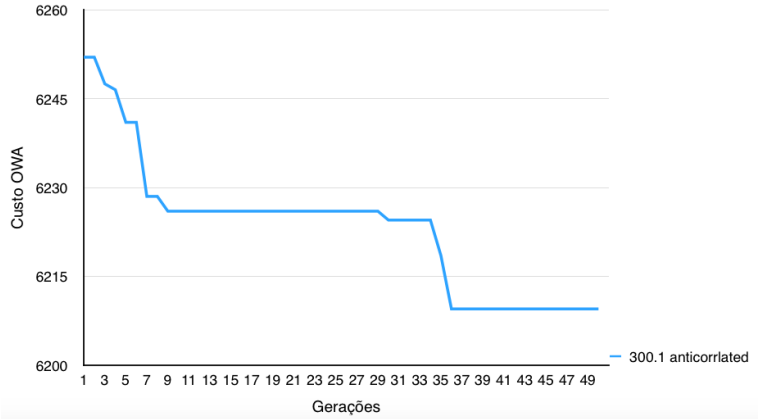


Figura 6: Evolução da solução de menor custo para a instância 300.1 anticorrlated



A Tabela 3 mostra a contribuição de cada operador ou procedimento utilizado pelo *M-SA*. Tanto para instâncias *correlated* quanto *anticorrelated*, o operador de *crossover* teve mais sucesso que o operador de mutação. Recorde que esta taxa de sucesso remete à quantidade de vezes em que o operador conseguiu produzir uma solução melhor que aquela que lhe foi entregue. De fato, era esperado que a taxa de sucesso do *crossover* fosse melhor que a de mutação, pois este último não visa intensificar a solução, mas apenas deslocá-la, tentando, pois, livrar a busca de ótimos locais. As Figuras 7 e 8 ilustram o sucesso de tais operadores em função do tamanho da instância, respectivamente, para as classes *correlated* e *anticorrelated*. Em ambas as classes, nota-se um equilíbrio entre a contribuição do *crossover* e do *SA*. Em média, este último teve a maior contribuição para instâncias pequenas. Contudo, mesmo para instâncias de grande porte, a contribuição do *SA* se manteve significativa, o que justifica seu uso. As taxas de sucesso do *crossover* e do *SA* oscilam bastante, e seu comportamento é irregular. Ademais, as instâncias destes experimentos apresentam apenas um critério OWA (*k-trimmed*). Novos experimentos, em outros critérios OWA, fazem-se necessários a fim de verificar a efetiva contribuição do *SA*.

Tabela 3: Estatísticas do *M-SA* em instâncias *k-trimmed* com 10 objetivos

Instância	Correlated					Anticorrelated				
	Média da taxa de sucesso do crossover	Média da taxa de sucesso da mutação	Média da taxa de sucesso do SA	Média da quantidade de renovações da população	Média da quantidade de vezes em que o fitness foi avaliado	Média da taxa de sucesso do crossover	Média da taxa de sucesso da mutação	Média da taxa de sucesso do SA	Média da quantidade de renovações da população	Média da quantidade de vezes em que o fitness foi avaliado
30.1	0,083	0,012	0,103	7,833	3,0962E+06	0,089	0,019	0,129	7,867	3,0962E+06
30.2	0,091	0,024	0,140	8,033	3,0962E+06	0,080	0,006	0,088	9,000	3,0963E+06
30.3	0,032	0,006	0,135	9,000	3,0963E+06	0,038	0,004	0,156	8,733	3,0962E+06
35.1	0,087	0,011	0,099	8,900	3,0962E+06	0,087	0,011	0,087	8,567	3,0962E+06
35.2	0,049	0,009	0,072	8,133	3,0962E+06	0,068	0,007	0,068	8,700	3,0962E+06
35.3	0,040	0,006	0,086	8,933	3,0962E+06	0,032	0,005	0,099	9,000	3,0963E+06
40.1	0,078	0,013	0,051	8,133	3,0962E+06	0,067	0,009	0,041	8,800	3,0962E+06
40.2	0,088	0,008	0,062	9,033	3,0963E+06	0,086	0,011	0,072	8,900	3,0962E+06
40.3	0,065	0,009	0,091	8,833	3,0962E+06	0,062	0,008	0,091	8,933	3,0962E+06
45.1	0,037	0,006	0,039	8,567	3,0962E+06	0,106	0,009	0,044	7,800	3,0961E+06
45.2	0,077	0,012	0,056	8,967	3,0963E+06	0,103	0,009	0,054	6,800	3,0961E+06
45.3	0,059	0,005	0,080	8,833	3,0962E+06	0,043	0,014	0,077	9,033	3,0963E+06
50.1	0,090	0,009	0,051	8,333	3,0962E+06	0,107	0,014	0,063	6,767	3,0961E+06
50.2	0,086	0,010	0,080	7,733	3,0961E+06	0,086	0,010	0,074	7,533	3,0961E+06
50.3	0,028	0,008	0,082	8,900	3,0962E+06	0,055	0,007	0,068	8,500	3,0962E+06
100.1	0,121	0,011	0,044	5,333	3,0959E+06	0,105	0,014	0,076	7,200	3,0961E+06
100.2	0,075	0,007	0,038	8,167	3,0962E+06	0,092	0,020	0,043	8,967	3,0963E+06
100.3	0,076	0,008	0,064	7,867	3,0962E+06	0,081	0,012	0,071	8,367	3,0962E+06
200.1	0,093	0,010	0,035	7,733	3,0961E+06	0,106	0,016	0,079	4,800	3,0959E+06
200.2	0,109	0,015	0,059	4,467	3,0959E+06	0,103	0,011	0,043	6,567	3,0961E+06

Continua na próxima página

Tabela 3 – continuação da página anterior

Instância	Correlated					Anticorrelated				
	Média da taxa de sucesso do crossover	Média da taxa de sucesso da mutação	Média da taxa de sucesso do SA	Média da quantidade de renovações da população	Média da quantidade de vezes em que o fitness foi avaliado	Média da taxa de sucesso do crossover	Média da taxa de sucesso da mutação	Média da taxa de sucesso do SA	Média da quantidade de renovações da população	Média da quantidade de vezes em que o fitness foi avaliado
200.3	0,081	0,011	0,055	8,100	3,0962E+06	0,048	0,011	0,048	8,533	3,0962E+06
300.1	0,096	0,012	0,072	5,200	3,0959E+06	0,116	0,015	0,067	6,100	3,0960E+06
300.2	0,104	0,015	0,074	6,233	3,0960E+06	0,118	0,023	0,067	4,467	3,0959E+06
300.3	0,090	0,010	0,057	7,867	3,0962E+06	0,098	0,014	0,051	8,700	3,0962E+06
400.1	0,107	0,011	0,041	6,933	3,0961E+06	0,115	0,018	0,084	4,367	3,0959E+06
400.2	0,054	0,005	0,033	8,833	3,0962E+06	0,113	0,011	0,039	5,933	3,0960E+06
400.3	0,076	0,011	0,046	8,900	3,0962E+06	0,105	0,014	0,053	4,833	3,0959E+06
500.1	0,098	0,011	0,034	7,267	3,0961E+06	0,117	0,014	0,047	4,167	3,0959E+06
500.2	0,107	0,014	0,040	5,800	3,0960E+06	0,119	0,016	0,039	3,300	3,0958E+06
500.3	0,096	0,009	0,042	8,900	3,0962E+06	0,096	0,010	0,048	8,367	3,0962E+06

Figura 7: Taxa de sucesso dos operadores *crossover*, mutação e *SA* na classe correlated

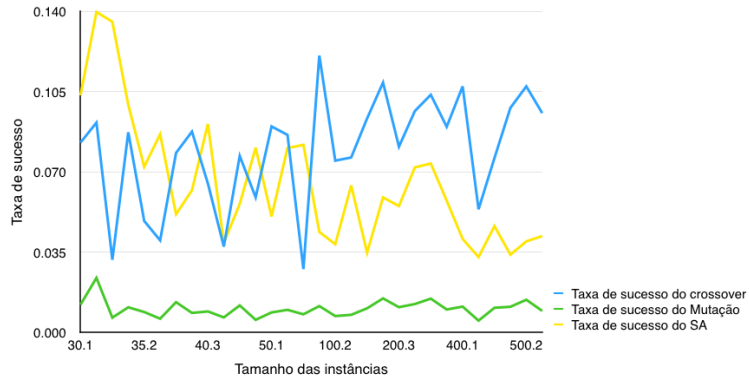
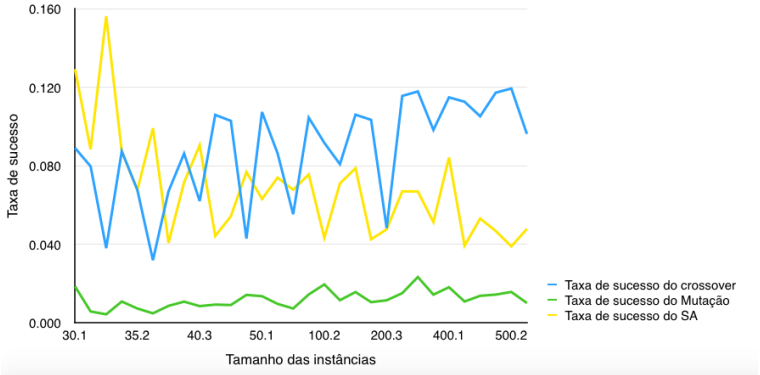
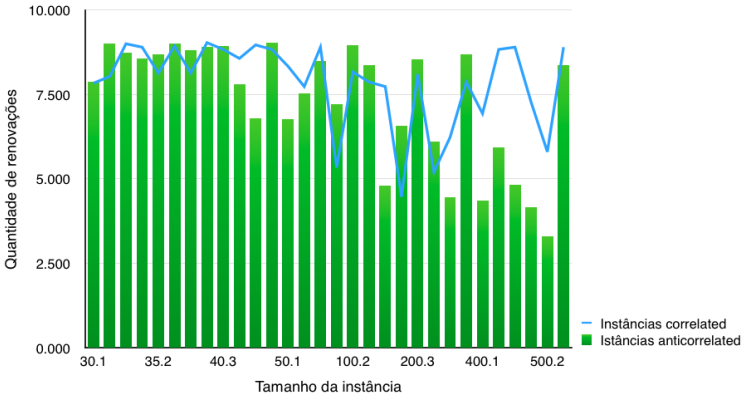


Figura 8: Taxa de sucesso dos operadores *crossover*, mutação e *SA* na classe anticorrelated



A Tabela 3 também mostra a quantidade média de renovações da população executadas pelo *M-SA*, conforme estratégia descrita na Seção 2.4. A Figura 9, em sintonia com a referida tabela, apresenta o comparativo entre as classes *correlated* e *anticorrelated*. A elevada quantidade média de renovações sugere que o ótimo corrente demora a evoluir. Conforme se discutiu nos parágrafos anteriores nesta seção, o processo evolucionário parece ter sido mais eficaz nas instâncias de grande porte da classe *anticorrelated*. Nelas, na maioria dos casos, principalmente para instâncias de grande porte, a quantidade de renovações foi menor e o ótimo corrente evoluiu mais. Faz-se, porém, necessários novos experimentos a fim de validar tal hipótese.

Figura 9: Média de renovações da população para instâncias de ambas as classes



Por fim, a Tabela 3 mostra a média, de 30 execuções, da quantidade de aviações da função *fitness*. À luz desta tabela, é possível afirmar que a variação desta quantidade, em função do tamanho da instância, é insignificante. Porém, seu valor é alto, em média $3.0962E+06$ para a classe *correlated* e $3.0961E+06$ para a classe *anticorrelated*.

4 Considerações finais

Este documento relatou os resultados experimentais de um algoritmo genético hibridizado com *simulated annealing* aplicado ao problema da *OWA-ST*. Experimentos mostraram que o algoritmo proposto obteve excelentes resultados face às soluções retornadas pelo *solver*, implementado com o modelo melhorado de Fernández et al. (2017), em até uma hora.

Foram analisados outros aspectos do algoritmo, como a contribuição de cada parte que o constitui. Os operadores *crossover*, mutação e o procedimento *SA* compartilham, de modo equilibrado, o trabalho da meta-heurística. Contudo, os operadores têm dificuldade de melhorar o ótimo corrente em algumas instâncias, principalmente na classe *correlated*. Por outro lado, na classe *anticorrelated*, a melhoria do ótimo corrente é levada até às últimas gerações. Isso pode ser confirmado pela quantidade de vezes em que o algoritmo necessitou renovar a população, a qual, na maioria dos casos, é menor nas instâncias *anticorrelated*.

Diante das limitações do algoritmo genético híbrido, como trabalhos futuros, será proposto um algoritmo transgenético baseado em operadores plasmídeos e transposons. A inspiração para tal proposta vem do trabalho de Monteiro, Goldbarg e Goldbarg (2010), onde o algoritmo transgenético proposto obteve os melhores resultados, em qualidade de solação e tempo de processamento, na literatura da Árvore Geradora Bi-objetivo. Ademais, ainda como trabalho futuro, almeja-se comparar o *M-SA* com o novo transgenético para o *OWA-ST*, procedendo experimentos para instâncias de até 1000 vértices e que adotem outros critérios OWA.

Referências

CHEN, G. et al. The multi-criteria minimum spanning tree problem based genetic algorithm. *Information Sciences*, v. 117, n. 22, p. 5050–5063, 2007.

FERNÁNDEZ, E. et al. Ordered weighted average optimization in multiobjective spanning tree problem. *European Journal of Operational Research*, Elsevier, v. 260, n. 3, p. 886–903, 2017.

GALAND, L.; SPANJAARD, O. Exact algorithms for owa-optimization in multiobjective spanning tree problems. *Computers & Operations Research*, Elsevier, v. 39, n. 7, p. 1540–1554, 2012.

HURWICZ, L. Optimality criteria for decision making under ignorance. *Cowles Commission Discussion Paper*, v. 370, p. 370, 1951.

JIANG, S.; YANG, S. A strength pareto evolutionary algorithm based on reference direction for multiobjective and many-objective optimization. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 21, n. 3, p. 329–346, 2017.

KNOWLES, J. *Local-search and hybrid evolutionary algorithms for Pareto optimization*. Tese (Doutorado) — Department of Computer Science, University of Reading, Reading, UK, 2002.

MONTEIRO, S. M. D.; GOLDBARG, E. F. G.; GOLDBARG, M. C. A new transgenetic approach for the biobjective spanning tree problem. In: *IEEE CEC 2010 Congress on Evolutionary Computation, 2010, Barcelona. Proceedings of IEEE CEC 2010 Congress on Evolutionary Computation.* [S.l.]: Piscataway, IEEE, 2010. v. 1, p. 519–526.

PRIM, R. C. Shortest connection networks and some generalizations. *Bell System Technical Journal*, v. 36, p. 1389–1401, 1957.

RAIDL, G. R.; JULSTROM, B. A. Edge sets: an effective evolutionary coding of spanning trees. *IEEE Transactions on Evolutionary Computation*, v. 7, n. 3, p. 225–239, 2003.

ROCHA, D. A. M.; GOLDBARG, E. F. G.; GOLDBARG, M. C. A memetic algorithm for the biobjective minimum spanning tree problem. In: *6th European Conference on Evolutionary Computation in Combinatorial Optimization, 2006. Budapeste, Lecture Notes in Computer Science*. [S.l.]: Heidelberg, Springer Berlin, 2006. v. 3906, p. 222–233.

TAMIR, A. The k-centrum multi-facility location problem. *Discrete Applied Mathematics*, Elsevier, v. 109, n. 3, p. 293–307, 2001.