Um Algoritmo Genético Adaptativo para o problema de sequenciamento em uma máquina com penalidades por antecipação e atraso da produção

Fábio Fernandes Ribeiro[1], **Marcone Jamilson Freitas Souza**[2], **Sérgio Ricardo de Souza**[1]

[1]Centro Federal de Educação Tecnológica de Minas Gerais
CEFET/MG – Belo Horizonte – MG – Brasil

[2]Universidade Federal de Ouro Preto - ICEB, Campus Universitário (UFOP)
CEP 35.400-000 – Ouro Preto – MG – Brasil

fabiobh@gmail.com, marcone@iceb.ufop.br, sergio@dppg.cefetmg.br

Abstract. *This paper treats the total tardiness single machine scheduling problem with earliness and tardiness penalties, considering due dates and sequence-dependent setup time. According to its complexity, an auto adaptive genetic algorithm is proposed to solve it. Many search operators are used to explore the solutions space where the choice probability for each operator depends to the success in a previous search. The initial population is generated by the combination between construct methods based on greedy, random and GRASP techniques. For each individual (job sequence) generated, a polynomial time algorithm are used to determine the processing initial optimal date to each job. Computational results show the effectiveness of the proposed algorithm.*

Resumo. *Este trabalho trata do problema de sequenciamento em uma máquina com penalidades por antecipação e atraso da produção, considerando janelas de entrega e tempo de preparação de máquina dependente da sequência de produção. Em vista da complexidade combinatória do problema, propõe-se resolvê-lo por meio de um algoritmo genético auto-adaptativo. Para cada indivíduo (sequência de tarefas) gerado utiliza-se um algoritmo de tempo polinomial para determinar a data ótima de início de processamento de cada tarefa na sequência dada. Cinco operadores de cruzamento são utilizados para uma melhor exploração do espaço de soluções, sendo que a probabilidade de escolha de cada um deles depende do sucesso em buscas pregressas. Testes computacionais mostram a efetividade do algoritmo proposto.*

1. Introdução

O problema de programação da produção objeto de estudo deste trabalho é relativo ao sequenciamento em uma máquina com minimização das penalidades por antecipação e atraso da produção (*Single Machine Scheduling for Minimizing Earliness and Tardiness Penalties*), denominado PSUMAA. No problema abordado, considera-se, também, a existência de janelas de entrega e tempos de preparação de máquina dependentes da sequência de produção. O objetivo é determinar a sequência na qual as tarefas serão processadas numa determinada máquina, durante um período de tempo, e o momento em que elas serão processadas, de modo a minimizar os custos por antecipação ou atraso na entrega.

Para a resolução deste problema, um Algoritmo Genético adaptativo é proposto. A população inicial deste algoritmo é gerada aplicando-se a fase de construção GRASP [Feo and Resende 1995], tendo como função guia cinco regras de despacho (TDD, EDD, SPT, WSPT e LPT). Para cada construção (GRASP + Regra de despacho), são gerados 200 indivíduos. Em seguida, estes são ordenados, do melhor para o pior, segundo a função de avaliação. A população inicial é, então, formada pelos 100 melhores indivíduos gerados. O cruzamento utiliza um método adaptativo baseado na qualidade dos indivíduos gerados por cada operador *crossover* ao longo da evolução, ou seja, após um certo número de gerações (no caso, cinco), o fator de escolha do operador a ser aplicado é atualizado de acordo com a qualidade dos indivíduos formados por cada um deles nas gerações anteriores. Um procedimento de busca local é, então, aplicado nesses melhores indivíduos de cada operador *crossover* com o objetivo de refinamento. A população sobrevivente é composta pelos cinco melhores indivíduos refinados desses operadores, pelo melhor indivíduo encontrado até então e por 89 indivíduos escolhidos por elitismo. Outros cinco indivíduos são escolhidos aleatoriamente da geração corrente e submetidos à mutação, por meio de realocação de tarefas, para assegurar a diversidade da população. A população evolui até que o critério de parada seja atingido. A Reconexão por Caminhos [Glover 1996] é também aplicada a cada cinco gerações, e foi implementada caminhando-se do melhor indivíduo produzido pelo algoritmo até então em direção ao melhor indivíduo produzido por cada operador *crossover*.

O restante deste trabalho está estruturado como segue. Na Seção 2 é feita uma apresentação de trabalhos relacionados. As características do problema estudado são detalhadas na Seção 3, tendo o modelo matemático adotado apresentado na Seção 4. Na Seção 5 é descrito o Algoritmo Genético Adaptativo desenvolvido para resolução do PSUMAA abordado. Na Seção 6 são apresentados e discutidos os resultados computacionais. A Seção 7 conclui o trabalho e aponta perspectivas futuras para melhoramento do algoritmo proposto.

2. Trabalhos relacionados

O PSUMAA pertence à classe de problemas NP-difíceis, uma vez que uma versão simplificada dele, o problema de sequenciamento que tem o atraso total como critério de otimização, é NP-difícil [Du and Leung 1990]. Versões simplificadas do problema abordado são estudadas por vários autores. [Hino et al. 2005] usam Busca Tabu e Algoritmos Genéticos para a resolução do problema com datas comuns de entrega. Tais algoritmos usam de propriedades da solução ótima do problema para explorar o espaço de soluções. [Lee and Choi 1995] utilizaram Algoritmos Genéticos (AG) para a resolução do problema com datas de entrega distintas. Neste último trabalho, um algoritmo específico, de complexidade polinomial foi desenvolvido para determinar a data ótima de conclusão de processamento de cada tarefa da sequência produzida pelo AG. Esse algoritmo é necessário porque pode valer a pena antecipar uma tarefa, mesmo pagando uma penalidade, se essa penalização for menor que a decorrente do atraso. [Wan and Yen 2002] trataram esse problema considerando janelas de entrega, ao invés de datas de entrega. É feita uma adaptação do algoritmo de [Lee and Choi 1995] para determinar as datas ótimas de cada tarefa, de forma a incluir essas novas características.

A utilização de Algoritmos Genéticos adaptativos, ou seja, aqueles que ajustam os parâmetros do algoritmo dinamicamente, pode ser uma maneira de otimizar a velocidade

de convergência de um algoritmo para o ótimo [Barcellos 2000]. Segundo [Matias 2008], o controle dos parâmetros e operadores dos Algoritmos Genéticos durante a sua execução é de fundamental importância, pois permite um ajuste automático em tempo de execução. Este autor propôs uma técnica de adaptação automática dos principais operadores dos Algoritmos Genéticos baseada no desempenho do algoritmo e na distribuição dos indivíduos da população no espaço de busca. A técnica estudada permite ao Algoritmo Genético ajustar o valor dos operadores, de forma a privilegiar aqueles que podem produzir resultados melhores em um determinado momento da busca.

3. O problema de sequenciamento abordado

O problema objeto deste trabalho é o de sequenciamento em uma máquina (PSUMAA), com tempo de preparação dependente da sequência de produção e janelas de entrega. Neste problema, considera-se que uma máquina deve processar um conjunto de n tarefas. Considera-se que cada tarefa possui um tempo de processamento P_i, uma data inicial E_i e uma data final T_i, desejadas para o término do processamento. Além disso, a executa no máximo uma tarefa por vez e, uma vez iniciado o processamento de uma tarefa, a mesma deverá ser finalizada, não sendo permitida a interrupção do processamento. Admite-se que todas as tarefas estejam disponíveis para processamento na data 0. Considera-se também que quando uma tarefa j é sequenciada imediatamente após uma tarefa i, sendo estes pertencentes a diferentes famílias de produtos, é necessário um tempo S_{ij} para a preparação da máquina. Tempos de preparação de máquina nulos ($S_{ij} = 0$) implicam em produtos da mesma família.

Considera-se, ainda, que a máquina não necessita de tempo de preparação inicial, ou seja, o tempo de preparação da máquina para o processamento da primeira tarefa na sequência é igual a 0. Permite-se, neste trabalho, tempo ocioso entre a execução de duas tarefas consecutivos. Considera-se, além disso, que tarefas devem ser finalizadas dentro da janela de tempo $[E_i, T_i]$, denominada janela de entrega. Caso a tarefa seja finalizada antes de E_i, há, então, uma penalização por antecipação. Caso a tarefa seja finalizada após T_i, então ocorrerá uma penalização por atraso. As tarefas que forem finalizadas dentro da janela de entrega não proporcionarão nenhum custo. Por fim, admite-se que custos unitários por antecipação e atraso da produção sejam dependentes das tarefas, ou seja, cada tarefa i possui um custo unitário de antecipação α_i e um custo unitário de atraso β_i. O objetivo do problema é a minimização do somatório dos custos de antecipação e atraso da produção.

4. Modelo de programação matemática

Apresenta-se, a seguir, o modelo de programação linear inteira mista (PLIM) para o PSUMAA, na forma proposta por [Gomes Jr. et al. 2007]. Sejam s_i a data de início do processamento da tarefa $i (s_i \geq 0)$; e_i o tempo de antecipação; e t_i o tempo de atraso da tarefa i. Sejam, também, duas tarefas fictícias, 0 (zero) e $n + 1$, de tal forma que 0 antecede imediatamente a primeira operação e $n + 1$ sucede imediatamente a última tarefa na sequência de produção. Admite-se que os tempos de processamento $P_0 = P_{n+1} = 0$ e que os tempos de preparação de máquina $S_{0i} = S_{i,n+1} = 0 \ \forall i = 1, \cdots, n$. Seja y_{ij} uma variável binária, que possui valor 1 se a tarefa j suceder imediatamente a tarefa i e valor 0, caso contrário. Considere, ainda, uma constante M de valor suficientemente grande. O modelo PLIM proposto pelos autores é apresentado a seguir:

$$\text{min} \quad Z = \sum_{i=1}^{n} (\alpha_i e_i + \beta_i t_i) \tag{1}$$

$$\text{s.a:} \quad s_j - s_i - y_{ij}(M + S_{ij}) \geq P_i - M \quad \forall\, i = 0, \ldots, n; \quad \forall\, j = 1, \ldots, n+1 \text{ e } i \neq j; \tag{2}$$

$$\sum_{j=1,\, j\neq i}^{n+1} y_{ij} = 1 \qquad \forall\, i = 0, \ldots, n \tag{3}$$

$$\sum_{i=0,\, i\neq j}^{n} y_{ij} = 1 \qquad \forall\, j = 1, \ldots, n+1 \tag{4}$$

$$s_i + P_i + e_i \geq E_i \qquad \forall\, i = 1, \ldots, n \tag{5}$$

$$s_i + P_i - t_i \leq T_i \qquad \forall\, i = 1, \ldots, n \tag{6}$$

$$s_i \geq 0 \qquad \forall\, i = 0, \ldots, n+1 \tag{7}$$

$$e_i \geq 0 \qquad \forall\, i = 1, \ldots, n \tag{8}$$

$$t_i \geq 0 \qquad \forall\, i = 1, \ldots, n \tag{9}$$

$$y_{ij} \in \{0, 1\} \qquad \forall\, i, j = 0, \ldots, n+1 \tag{10}$$

A função objetivo, representada pela equação (1), tem, como critério de otimização, a minimização dos custos de antecipação e atraso. As restrições (2) definem a sequência de operações sobre o recurso (máquina) utilizado, ou seja, elas garantem que haja tempo suficiente para completar uma tarefa i antes de começar uma tarefa j. As restrições (3) e (4) garantem que cada tarefa tenha somente uma tarefa imediatamente antecessora e uma tarefa imediatamente sucessora, respectivamente. As restrições (5) e (6) computam os valores do atraso e da antecipação de acordo com a janela de entrega desejada para o término do processamento de cada tarefa. As restrições (7) a (10) definem o tipo das variáveis do problema.

5. Metodologia heurística

Nesta seção descreve-se o Algoritmo Genético Adaptativo proposto para resolver o PSU-MAA abordado.

5.1. Representação de um indivíduo

Um indivíduo (sequência de tarefas) é representado por um vetor v de n genes (tarefas). A posição de cada gene indica sua ordem de produção. Por exemplo, no indivíduo $v = \{7, 1, 5, 6, 4, 3, 2\}$, a tarefa 7 é a primeira a ser processada e a tarefa 2, a última.

5.2. Avaliação dos indivíduos

Todos os indivíduos da população são avaliados pela própria função objetivo, dada pela equação (1) do modelo de PLIM, onde são considerados mais adaptados aqueles que obtiverem o menor valor.

5.3. Construção de uma população inicial

A população inicial do Algoritmo Genético Adaptativo proposto é gerada aplicando-se a fase de construção GRASP ([Feo and Resende 1995]) tendo como função guia cinco

regras de despacho (EDD, TDD, SPT, WSPT e LPT). Para cada construção (GRASP + Regra de despacho) são gerados 200 indivíduos. Em seguida, estes são ordenados, do melhor para o pior segundo a função de avaliação. A população inicial é, então, composta pelos 100 melhores indivíduos gerados.

5.3.1. Regras de despacho

Na regra de despacho EDD (*Earliest Due Date*), as tarefas são ordenadas tendo em vista a data de início da janela de entrega. As tarefas com datas de início mais próximas são processadas antes daquelas com datas de início maiores. Pela regra de despacho TDD (*Tardiness Due Date*), as tarefas também são ordenadas com base na data de início da janela de entrega, porém aquelas com datas de início mais tardias são processadas antes daquelas com datas de início mais cedo. A regra de despacho SPT (*Shortest Processing Time*) constroi a sequência de tarefas ordenando-as de tal forma que a tarefa com tempo de processamento mais curto seja processada antes daquela com tempo de processamento mais longo. Na regra WSPT (*Weight Shortest Processing Time*) utiliza-se basicamente a mesma lógica do SPT, porém levando-se em consideração um peso que é atribuído a cada tarefa, em função da sua prioridade de atendimento. As tarefas são então ordenadas a partir da ordem crescente ponderada da razão entre os tempos de processamento e a sua prioridade de processamento. Finalmente, a regra de despacho LPT (*Longest Processing Time*) elabora a sequência de tarefas criando uma ordem tal que a tarefa com tempo de processamento mais longo seja processada antes daquela com o tempo de processamento mais curto, ou seja, as tarefas com processamento mais longo são as primeiros a serem processadas.

5.3.2. Fase de construção GRASP

O procedimento de construção segue as idéias da fase de construção do algoritmo GRASP [Feo and Resende 1995]. Neste procedimento, um indivíduo é formado gene a gene, de acordo com um critério g de seleção. Para estimar o benefício da inserção de cada gene (tarefa), utiliza-se uma das regras de despacho descritas na seção anterior, regra essa que é escolhida de forma aleatória, mas é fixa durante toda a construção. A cada inserção, os próximos genes candidatos a formarem o indivíduo são colocados em uma lista de candidatos, respeitando o critério de seleção. Os melhores candidatos, são então colocados em uma Lista Restrita de Candidatos (LRC). A seguir, um desses candidatos é escolhido randomicamente e colocado no indivíduo em construção. O procedimento é encerrado quando todos os genes são alocados, situação na qual o indivíduo está completamente formado. A Figura 1 mostra o pseudocódigo dessa fase de construção. Nesta figura, g_{min} representa o melhor valor assumido pela Regra de Despacho escolhida e g_{max}, o maior.

Um parâmetro γ, com $0 \leq \gamma \leq 1$, é utilizado para controlar o tamanho da LRC. Empiricamente, foram gerados preliminarmente 10.000 indivíduos para cada um dos seguintes valores de γ: $0,02; 0,04; 0,06; 0,08; 0,10; 0,12; 0,14; 0,16; 0,18; 0,20$. Tendo-se como base um problema-teste envolvendo 20 tarefas e as cinco regras de despacho, cada qual escolhida aleatoriamente em cada execução, o melhor comportamento foi observado no parâmetro $\gamma = 0,20$.

procedimento $Construcao(g(.), \gamma, v)$;
1 $v \leftarrow \emptyset$;
2 Inicialize o conjunto C de genes candidatos;
3 enquanto $(C \neq \emptyset)$ faça
4 $g_{min} = \min\{g(t) \mid t \in C\}$;
5 $g_{max} = \max\{g(t) \mid t \in C\}$;
6 $LRC = \{t \in C \mid g(t) \leq g_{min} + \gamma(g_{max} - g_{min})\}$;
7 Selecione, aleatoriamente, um gene $t \in LRC$;
8 $v \leftarrow v \cup \{t\}$;
9 Atualize o conjunto C de genes candidatos;
10 fim-enquanto;
11 Retorne v;
fim $Construcao$;

Figura 1. Procedimento para construir um indivíduo

Com base nestas informações, optou-se por fixar γ em 0,20, por se ter encontrado o menor valor para a função de avaliação, apesar de o *gap* ter sido o maior.

5.4. Algoritmo genético adaptativo aplicado ao PSUMAA

A Figura 2 apresenta o pseudocódigo do Algoritmo Genético Adaptativo (AGA) proposto. As fases desse algoritmo são, a seguir, detalhadas.

Método de seleção de indivíduos: Após a avaliação de toda a população, os indivíduos são selecionados por meio do método da roleta, cujo principal objetivo é permitir que os indivíduos mais adaptados tenham maior probabilidade de serem selecionados.

Cruzamento: Após a avaliação de toda a população, os indivíduos são selecionados para reprodução através do método de seleção de indivíduos já descrito. Os operadores *crossover* utilizados são o *One Point Crossover* (OX), o *Similar Job Order Crossover* (SJOX), o *Relative Job Order Crossover* (RRX), o *Based Order Uniform Crossover* (BOUX) e o *Partially Mapped Crossover* (PMX) por apresentarem alta eficiência para o PSUMAA [Lee and Choi 1995]. A probabilidade de escolha de um operador *crossover* específico varia de acordo com a qualidade das soluções produzidas em gerações passadas. Mais precisamente, sejam O_i, com $i = 1, \cdots, 5$, os cinco operadores *crossover*. Inicialmente, cada operador *crossover* O_i tem a mesma probabilidade de ser escolhido, no caso, $p(O_i) = 1/5$. Seja $f(s*)$ a melhor solução encontrada até então e A_i o valor médio das soluções encontradas por cada operador O_i desde a última atualização. Caso o operador não tenha sido escolhido nas últimas cinco gerações, faz-se $A_i = 1$. Seja $q_i = f(s^*) = A_i$ e $p(O_i) = q_i / \sum_{j=1}^{5} q_j$ onde $i = 1, \cdots, 5$. Observe que quanto melhor a solução, maior o valor de q_i e, consequentemente, maior a probabilidade $p(O_i)$ de se escolher o operador O_i. Portanto, durante a evolução do algoritmo, o melhor operador tem sua chance de escolha incrementada. Este procedimento foi inspirado no algoritmo *Reactive GRASP*, proposto por [Prais and Ribeiro 2000].

Busca Local: Como dito anteriormente, a cada cinco gerações é aplicada busca local ao melhor indivíduo gerado por cada operador *crossover*. O método de busca local aplicado é a Descida Randômica, a qual usa dois tipos de movimento para explorar o espaço de busca: a troca da ordem de processamento de duas tarefas da sequência de produção e a realocação de uma tarefa para outra posição na sequência de produção. O método

```
Algoritmo AGA(maxger, numind, probcrossover, probmutacao);
1  t ← 0;
2  Gere a população inicial P(t);
3  Avalie P(t);
4  enquanto (t ≤ maxger) faça
5      t ← t + 1;
6      Gere P(t) a partir de P(t − 1);
7      enquanto (i ≤ numind) faça
8          i ← 1;
9          cross ← Número aleatório de 1 a 100;
10         se (cross ≤ probcrossover) faça
11             Selecionar indivíduo;
12             Cruzar;
13         fim-se;
14         Avaliar P(t);
15     fim-enquanto (i ≤ numind) faça
16     Definir a população sobrevivente;
17     se (t mod 5 = 0) faça
18         Atualizar probabilidade de escolha de cada operador (p(O_i));
19         Executar Busca local;
20         Aplicar Reconexão por Caminhos;
21     fim-se;
22 fim-enquanto;
fim AGA;
```

Figura 2. Pseudocódigo do Algoritmo Genético Adaptativo

funciona como segue. Para uma dada solução, seleciona-se aleatoriamente duas tarefas, trocando-as de posição. Se esse novo vizinho for melhor que o anterior segundo a função de avaliação, ele é aceito e passa a ser a solução corrente; caso contrário, outro vizinho é escolhido aleatoriamente. Se durante *MRDmax* nenhuma melhor solução for gerada, então passa-se a usar movimentos de realocação. Havendo melhora com este tipo de movimento, retorna-se à utilização de movimentos de troca; caso contrário, encerra-se a busca local decorridas *MRDmax* iterações sem melhora.

Reconexão por Caminhos: A Reconexão por Caminhos (*Path Relinking*) é um procedimento que integra estratégias de intensificação e diversificação durante o processo de busca [Rossetti 2003]. Ele gera novas soluções explorando trajetórias que ligam soluções de alta qualidade. Dado um par de soluções, começa-se a busca com uma delas, dita solução base, e chega-se à outra, dita solução guia, adicionando-se gradualmente atributos da solução guia à solução base. No problema em questão, durante o processo evolutivo forma-se um grupo com os cinco melhores indivíduos obtidos por cada operador *crossover*. Periodicamente(a cada cinco gerações), aciona-se a aplicação da Reconexão por Caminhos tendo-se como solução base o melhor indivíduo encontrado pelo algoritmo até então e como solução guia, cada um dos melhores indivíduos formados por cada operador *crossover*. O procedimento aplicado é a Reconexão por Caminhos Regressiva Truncada (*Truncated Backward Path Relinking*), no qual interrompe-se a busca quando 75% dos

atributos da solução guia forem inseridos na solução base. Considera-se como atributo a posição da tarefa na sequência de produção. Para cada tarefa candidata à inserção, aplica-se o módulo de Busca Local descrito anteriormente, não permitindo a movimentação da tarefa candidata. A tarefa efetivamente inserida é aquela que produzir o melhor valor para a função de avaliação.

Sobrevivência de indivíduos: A sobrevivência de indivíduos da população é aplicada utilizando-se a técnica de elitismo. Sobrevivem 95% dos indivíduos mais adaptados e os 5% restantes são compostos por indivíduos retirados aleatoriamente da geração corrente, sendo estes submetidos à mutação por troca entre duas tarefas.

Critério de parada: Como critério de parada, adota-se o número máximo de gerações.

6. Resultados computacionais

O Algoritmo Genético Adaptativo proposto foi implementado em linguagem C, utilizando o ambiente C++ Builder 5. Seus parâmetros, obtidos experimentalmente, são apresentados na Tabela 1.

Tabela 1. Parâmetros do Algoritmo Genético Adaptativo

Parâmetros	Valores
Parâmetro γ da fase de construção GRASP	0,20
Número máximo de iterações da Busca Local (*MRDmax*)	$7 \times n$
Número máximo de gerações do AG Adaptativo (*maxger*)	100
Probabilidade de cruzamento	80%
Taxa de mutação	5%

Os testes computacionais foram realizados em um computador Pentium Core 2 Duo 2,1 GHz, com 4 GB de memória RAM, sob plataforma Windows Vista. Os problemas-teste utilizados são os de [Gomes Jr. et al. 2007] e envolvem número de tarefas n igual a 8, 9, 10, 11, 12, 15, 20, 25, 30, 40, 50 e 75. São 12 problemas-teste para cada número de tarefas, totalizando 144 problemas-teste. Cada problema-teste foi resolvido 30 vezes pelo método proposto, com exceção daqueles envolvendo 75 tarefas, os quais foram resolvidos apenas 20 vezes, pois demandavam um tempo computacional mais elevado. Para os problemas-teste de 75 tarefas adotou-se, ainda, o valor *MRDmax* = $2 \times n$ como número máximo de iterações sem melhora da busca local para reduzir o tempo de processamento do algoritmo.

A Tabela 2 mostra os resultados obtidos para o Algoritmo Genético Adaptativo proposto, bem como reproduz os de [Gomes Jr. et al. 2007]. A primeira coluna mostra o número de tarefas envolvidas. Na segunda e terceira colunas é mostrado o quanto as soluções médias de cada algoritmo (AGA e [Gomes Jr. et al. 2007], respectivamente) desviaram da melhor solução conhecida referente a cada problema-teste. Já nas duas últimas colunas é mostrado o quanto as melhores soluções geradas por tais algoritmos desviaram das melhores soluções conhecidas. O *Desvio* percentual é calculado pela expressão $Desvio = [(RMed - MR)/MR] \times 100\%$, sendo *RMed* o resultado médio obtido pela aplicação do respectivo algoritmo e *MR*, o melhor resultado conhecido de cada problema-teste até então.

Tabela 2. Resultados do Algoritmo Genético Adaptativo

#	*Desvio* da solução média		*Desvio* da melhor solução	
Tarefas	AGA	Gomes Jr.	AGA	Gomes Jr.
8	0,00	0,03	0,00	0,00
9	0,15	0,06	0,00	0,00
10	0,24	0,02	0,00	0,00
11	0,03	0,12	0,00	0,00
12	0,07	0,21	0,00	0,00
15	0,76	1,47	0,00	0,00
20	0,73	1,65	0,00	0,00
25	1,02	2,32	0,00	0,00
30	1,60	3,34	0,00	0,20
40	2,15	4,38	-0,25	0,18
50	3,72	6,13	-0,60	0,28
75	4,59	10,89	-1,48	0,56

Como pode ser observado, para problemas-teste envolvendo acima de onze tarefas, os desvios das soluções médias do algoritmo proposto foram sempre menores que aqueles de [Gomes Jr. et al. 2007]. Esta capacidade de produzir soluções finais com uma menor variabilidade mostra uma maior robustez do AGA quando comparado com esse algoritmo da literatura. Por outro lado, o AGA também é capaz não somente de gerar as melhores soluções conhecidas, mas também de produzir soluções ainda melhores. De fato, o AGA foi capaz de gerar todas as melhores soluções conhecidas para problemas até 30 tarefas, ao contrário do algoritmo de [Gomes Jr. et al. 2007], que gerou todas as melhores soluções somente até problemas com 25 tarefas, ficando a 0,20% do melhor valor conhecido para problemas envolvendo 30 tarefas. Finalmente, para os problemas-teste envolvendo 40, 50 e 75 tarefas, o AGA foi capaz de superar as melhores soluções conhecidas em até 1,48%.

7. Conclusões

Este artigo teve seu foco no PSUMAA considerando janelas de entrega e tempo de preparação da máquina dependente da sequência de produção. Para resolvê-lo foi proposto um Algoritmo Genético Adaptativo, onde a população inicial foi gerada por um procedimento GRASP, utilizando como função guia as regras de despacho EDD (*Earliest Due Date*), TDD (*Tardiness Due Date*), SPT (*Shortest Processing Time*), WSPT (*Weight Shortest Processing Time*) e LPT (*Longest Processing Time*). Durante o processo evolutivo, a população passa pelas fases de seleção, cruzamento e mutação. No cruzamento, cinco operadores *crossover*, OX (*One Point Crossover*), SJOX (*Similar Job Order Crossover*), BOUX (*Based Order Uniform Crossove*), PMX (*Partially Mapped Crossover*) e RRX (*Relative Job Order Crossover*), são utilizados, sendo que a escolha de qual operador será empregado depende da qualidade das soluções produzidas em gerações passadas. Periodicamente, as melhores soluções produzidas por cada operador *crossover* são submetidas à busca local e à Reconexão por Caminhos. O procedimento de Reconexão liga a melhor solução produzida até então a cada uma das melhores soluções de cada operador.

Para testá-lo, foram utilizados problemas-teste da literatura, sendo o mesmo comparado com um algoritmo da literatura. Nos problemas envolvendo até 30 tarefas, o

algoritmo proposto apresentou soluções de alta qualidade e com baixo desvio, sempre atingindo o ótimo. Já em problemas de dimensões maiores (40 a 75 tarefas), o algoritmo desenvolvido apresentou soluções melhores que as desse algoritmo da literatura, além de apresentar uma menor variabilidade das soluções finais, mostrando sua robustez.

Agradecimentos

Os autores agradecem ao CEFET-MG, CAPES e FAPERJ pelo apoio ao desenvolvimento deste trabalho.

Referências

Barcellos, J. C. H. d. (2000). "Algoritmos genéticos adaptativos: Um estudo comparativo". Dissertação de mestrado, Escola Politécnica da USP, São Paulo.

Du, J. and Leung, J. Y. T. (1990). "Minimizing total tardiness on one machine is np-hard". *Mathematics of Operations Research*, 15:483–495.

Feo, T. A. and Resende, M. G. C. (1995). "Greedy randomized adaptive search procedures". *Journal of Global Optimization*, 6:109–133.

Glover, F. (1996). "Tabu search and adaptive memory programming - advances, applications and challenges". In Barr, R. S., Helgason, R. V., and Kennington, J. L., editors, *Computing Tools for Modeling, Optimization and Simulation: Interfaces in Computer Science and Operations Research*, pages 1–75. Kluwer Academic Publishers.

Gomes Jr., A. C., Carvalho, C. R. V., Munhoz, P. L. A., and Souza, M. J. F. (2007). "A hybrid heuristic method for solving the single machine scheduling problem with earliness and tardiness penalties (in portuguese)". In *Proceedings of the XXXIX Brazilian Symposium of Operational Research (XXXIX SBPO)*, pages 1649–1660, Fortaleza, Brazil.

Hino, C. M., Ronconi, D. P., and Mendes, A. B. (2005). "Minimizing earliness and tardiness penalties in a single-machine problem with a common due date". *European Journal of Operational Research*, 160:190–201.

Lee, C. Y. and Choi, J. Y. (1995). "A genetic algorithm for job sequencing problems with distinct due dates and general early-tardy penalty weights". *Computers and Operations Research*, 22:857–869.

Matias, P. T. (2008). "Avaliação comparativa de algoritmos evolutivos com operadores adaptativos". Dissertação de mestrado, Programa de Engenharia Civil, UFRJ, Rio de Janeiro.

Prais, M. and Ribeiro, C. C. (2000). "An application to a matrix decomposition problem in tdma traffic assignment". *INFORMS - Journal on Computing*, 12:164–176.

Rossetti, I. C. M. (2003). *"Estratégias sequenciais e paralelas de GRASP com reconexão por caminhos para o problema de síntese de redes a 2 caminhos"*. Tese de doutorado, Programa de Pós-Graduação em Informática, PUC-RJ, Rio de Janeiro, Brasil.

Wan, G. and Yen, B. P. C. (2002). "Tabu search for single machine scheduling with distinct due windows and weighted earliness/tardiness penalties". *European Journal of Operational Research*, 142:271–281.