RESOLUÇÃO DO PROBLEMA DE SEQUENCIAMENTO EM UMA MÁQUINA COM MINIMIZAÇÃO DAS PENALIDADES POR ANTECIPAÇÃO E ATRASO DA PRODUÇÃO ATRAVÉS DE ALGORITMO GENÉTICO ADAPTATIVO

Fábio Fernandes Ribeiro (CEFET-MG)
fabiobh@gmail.com
Sérgio Ricardo de Souza (CEFET-MG)
sergio@dppg.cefetmg.br
Marcone Jamilson Freitas Souza (UFOP)
marcone.freitas@oi.com.br

Neste trabalho, propõe-se um Algoritmo Genético Adaptativo para a resolução do Problema de Sequenciamento em uma Máquina com Minimização das Penalidades por Antecipação e Atraso da Produção (Single Machine Scheduling for Minimizing Earlineess and Tardiness Penalties), denominado PSUMAA, com janelas de entrega e tempos de preparação de máquina dependentes da sequência de produção. O algoritmo proposto usa a fase da construção da metaheuristica Greedy Randomized Adaptive Search Procedure (GRASP) para a construção da população inicial. Para a formação de novos indivíduos são usados cinco diferentes operadores de busca, os quais são aplicados, inicialmente, com uma mesma probabilidade. No entanto, durante o processo evolutivo, esta probabilidade muda de acordo com o desempenho do operador, de forma que os melhores recebem uma probabilidade maior. Durante o processo evolutivo, também é criado um grupo elite, formado por soluções de qualidade criadas por cada operador ao longo das gerações recentes. Soluções deste grupo são, então, conectadas à melhor solução gerada até então por meio do mecanismo Reconexão por Caminhos Truncada Regressiva (Truncated Backward Path Relinking). Os resultados computacionais apreentados mostram o sucesso da metodologia proposta.

Palavras-chaves: Sequenciamento em uma Máquina, Algoritmo Genético Adaptativo, Metaheurística

1. Introdução

Dentre os principais problemas de programação de produção existentes, destaca-se o de sequenciamento em uma máquina com penalidades por antecipação e atraso da produção. Segundo França Filho (2007), o estudo desses problemas é mais recente que aquele voltado para problemas em que o objetivo envolve uma função não-decrescente do instante de conclusão do processamento da tarefa, tais como tempo médio de fluxo, soma ponderada de atrasos e *makespan* (momento em que termina a execução da última tarefa). Para estes, os custos mais elevados decorrem apenas do adiamento da conclusão das tarefas. Entretanto, com a filosofia *just-in-time* adotada por muitas empresas, o foco atual é penalizar também a conclusão das tarefas antes do instante em que elas são requeridas. Isto é justicado pelo fato de que concluir uma tarefa antecipadamente pode resultar em custos financeiros extras pela necessidade antecipada de capital e/ou espaço para armazenamento e/ou de outros recursos para manter e gerenciar o estoque.

Uma restrição tecnológica significativa deste tipo de problema diz respeito ao tempo necessário à preparação da máquina para executar uma nova tarefa, conhecido como tempo de *setup*. Embora esse tempo apareça em vários processos produtivos, de acordo com Gupta e Smith (2006), muitas pesquisas desconsideram este tempo ou, então, acrescentam-no ao tempo de processamento da tarefa. Esta prática possibilita uma simplificação do problema, porém, compromete a qualidade das soluções quando existe uma grande variabilidade deste tempo, em função da ordenação das tarefas na máquina.

Problemas de sequenciamento em uma máquina com minimização do atraso total são NP-difíceis (Du e Leung, 1990); assim, também o são os problemas de sequenciamento com as características apontadas. Em vista da dificuldade de obtenção de soluções ótimas para esses problemas quando o número de tarefas a processar é elevado, abordagens heurísticas têm sido largamente utilizadas.

Neste trabalho, propõe-se um Algoritmo Genético Adaptativo para a resolução desta classe de problemas de sequenciamento, referenciada como problema de 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, com janelas de entrega e tempos de preparação de máquina dependentes da sequência de produção. O algoritmo proposto usa a fase da construção da metaheuristica *Greedy Randomized Adaptive Search Procedure* (GRASP) (Feo e Resende, 1995) para a construção da população inicial. Para a formação de novos indivíduos são usados cinco diferentes operadores de busca, os quais são aplicados, inicialmente, com uma mesma probabilidade. No entanto, durante o processo evolutivo, esta probabilidade muda de acordo com o desempenho do operador, de forma que os melhores recebem uma probabilidade maior. Durante o processo evolutivo, também é criado um grupo elite, formado por soluções de qualidade criadas por cada operador ao longo das gerações recentes. Soluções deste grupo são, então, conectadas à melhor solução gerada até então por meio do mecanismo Reconexão por Caminhos Truncada Regressiva (*Truncated Backward Path Relinking*).

Este trabalho está estruturado como se segue. Na Seção 2 é feita uma apresentação de trabalhos relacionados ao problema. As características do problema estudado são detalhadas na Seção 3 e na seção 4 o modelo matemático para o problema abordado é apresentado. Já na Seção 5 a metodologia proposta para resolução do PSUMAA é descrita. 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

Os problemas de sequenciamento estão entre os mais estudados na atualidade. Allahverdi *et al* (1999) ressaltam a importância desta classe de problemas, visto o grande número de aplicações práticas na indústria. Bustamante (2006), por exemplo, apresenta um exemplo de aplicação do PSUMAA em indústrias siderúrgicas. O conjunto de laminadores é considerado como uma única máquina que representa o gargalo do sistema e cada tarefa representa um conjunto de operações realizadas nesses laminadores, que produzem chapas com uma determinada espessura ou um fio metálico com um determinado diâmetro. Para cada produto (chapa ou fio) fabricado com espessura ou diâmetro diferente, têm-se um conjunto de operações de mesma natureza realizadas nos mesmos laminadores, mas que requerem tempos de processamento diferentes em cada um deles, configurando tarefas diferentes.

Ying (2008) resolveu esse problema considerando datas comuns de entrega das tarefas e tempo de *setup* de cada tarefa incluído em seu tempo de processamento e independente da sequência de produção. Foi utilizado o algoritmo *Recovering Beam Search* (RBS), versão aperfeiçoada do algoritmo *Beam Search* (BS), que é um algoritmo *branch-and-bound* no qual somente os *w* nós mais promissores de cada nível da árvore de busca são retidos para ramificação futura, enquanto os nós restantes são podados permanentemente. Com o objetivo de evitar que decisões erradas sobre poda de nós sejam tomadas, o algoritmo RBS utiliza uma fase de recobrimento, que busca por soluções parciais melhores que dominem aquelas selecionadas anteriormente.

Já Hino *et al* (2005) utilizaram Busca Tabu e Algoritmos Genéticos para a resolução do problema com datas comuns de entrega. Estes algoritmos usam de propriedades da solução ótima do problema para explorar o espaço de soluções.

Uma metodologia baseada na metaheurística Algoritmos Meméticos aplicada ao problema de sequenciamento de uma máquina com minimização do tempo total de atraso e tempo de preparação de máquina dependente da sequência de produção foi apresentada por França Filho (2007). O autor aplicou esta metodologia a instâncias com 20, 30, 40, 60, 80 e 100 tarefas. Os resultados são comparados com aqueles obtidos utilizando-se métodos baseados em AG. Os métodos baseados em algoritmos meméticos se mostraram superiores aos obtidos pelos métodos baseados em AG.

No trabalho de Gomes Jr. *et al.* (2007), um modelo de programação linear inteira mista (PLIM) para o PSUMAA com janelas de entrega e tempo de preparação dependente da sequência de produção foi desenvolvido, sendo que este modelo foi utilizado para resolver na otimalidade problemas de até 12 tarefas. A partir desta modelagem, os resultados obtidos por um método heurístico baseado em GRASP, *Iterated Local Search* (ILS) e *Variable Neighborhood Descent* (VND), também proposto pelos autores, foram comparados. Para cada sequência gerada pela heurística, é acionado um algoritmo, adaptado de Wan e Yen (2002), para determinar a data ótima de conclusão do processamento de cada tarefa. Este algoritmo inclui, no tempo de processamento de uma tarefa, o tempo de preparação da máquina, já que, quando ele é acionado, a sequência de produção é conhecida. Este algoritmo encontrou todas as soluções ótimas conhecidas em problemas de até 12 tarefas.

Em Lee e Choi (1995) foram utilizados Algoritmos Genéticos (AG) para a resolução do problema com datas de entrega distintas. Neste 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.

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. A avaliação da eficiência da técnica estudada é feita através de testes comparativos em funções *benchmark*, assim como numa aplicação real de engenharia que trata da otimização de sistemas de *risers* utilizados em exploração de petróleo *offshore*.

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 máquina 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 estas 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. Permite-se, neste trabalho, tempo ocioso entre a execução de duas tarefas consecutivas. 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, ocorrerá uma penalização por atraso. As tarefas que forem finalizadas dentro da janela de entrega não proporcionam 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. Modelagem 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 da tarefa i e t_i o tempo de atraso da tarefa i. Diferentemente de Bustamante (2006), foram utilizadas duas tarefas fictícias, 0 (zero) e n+1, de tal forma que a tarefa 0 antecede imediatamente a primeira operação e a tarefa n+1 sucede imediatamente a última tarefa na

sequência de produção. Admite-se que P_0 e P_{n+1} são iguais a zero e que S_{0i} =0 e $S_{i,n+1}$=0, $\forall i$ = 1, ..., n.

Seja y_{ij} uma variável binária definida da seguinte forma:

$$y_{ij} = \begin{cases} 1, & \text{se a tarefa } j \text{ é sequenciada imediatamente após a tarefa } i; \\ 0, & \text{caso contrário.} \end{cases}$$

Considere, ainda, uma constante M de valor suficientemente grande. O modelo de PLIM proposto de Gomes Jr. *et al*. (2007) é apresentado a seguir:

$$\text{minimizar} \quad Z = \sum_{i=1}^{n} (\alpha_i e_i + \beta_i t_i) \tag{3.1}$$

$$\text{sujeito a:} \quad s_j - s_i - y_{ij}(M + S_{ij}) \geq P_i - M \quad \forall\, i = 0,1,..., n; \quad \forall\, j = 1, 2, ..., n+1 \text{ e } i \neq j \tag{3.2}$$

$$\sum_{j=1, j\neq i}^{n+1} y_{ij} = 1 \quad \forall\, i = 0,1,..., n \tag{3.3}$$

$$\sum_{i=0, i\neq j}^{n} y_{ij} = 1 \quad \forall\, j = 1, 2, ..., n+1 \tag{3.4}$$

$$s_i + P_i + e_i \geq E_i \quad \forall\, i = 1, 2, ..., n \tag{3.5}$$

$$s_i + P_i - t_i \leq T_i \quad \forall\, i = 1, 2, ..., n \tag{3.6}$$

$$s_i \geq 0 \quad \forall i = 0, 1, ..., n+1 \tag{3.7}$$

$$e_i \geq 0 \quad \forall\, i = 1, 2, ..., n \tag{3.8}$$

$$t_i \geq 0 \quad \forall\, i = 1, 2, ..., n \tag{3.9}$$

$$y_{ij} \in \{0,1\} \quad \forall i,\ j = 0, 1, ..., n+1 \tag{3.10}$$

A função objetivo, representada pela equação (3.1), tem, como critério de otimização, a minimização dos custos de antecipação e atraso. As restrições (3.2) definem a sequência de operações sobre o recurso (máquina) utilizado, ou seja, garantem que haja um tempo suficiente para completar uma tarefa *i*, antes de começar uma tarefa *j*. As restrições (3.3) e (3.4) garantem que cada tarefa tenha somente uma tarefa imediatamente antecessora e uma tarefa imediatamente sucessora, respectivamente. As restrições (3.5) e (3.6) definem os valores do atraso e da antecipação, de acordo com a janela de entrega desejada para o término do processamento da tarefa *i*, caso estes existam. As restrições de (3.7) a (3.10) definem o tipo das variáveis do problema.

5. Metodologia Heurística

Nesta seção, descreve-se o método adaptativo proposto para resolver o PSUMAA 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.

5.2 Avaliação dos indivíduos

Todos os indivíduos da população são avaliados pela própria função objetivo, dada pela equação (3.1) do modelo de PLIM, em que são considerados mais adaptados aqueles indivíduos que obtiverem o menor valor.

5.3 Construção de uma população inicial

A população inicial do método adaptativo proposto é gerada aplicando-se a fase de construção GRASP (FEO e 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 Fase de construção GRASP

O procedimento de construção segue as idéias da fase de construção do algoritmo GRASP (FEO E 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, 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-se 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 inserido no indivíduo em construção. O procedimento é encerrado quando todos os genes são alocados, situação na qual o indivíduo está completamente formado.

5.4 Método adaptativo aplicado ao PSUMAA

A Figura 2 apresenta o pseudocódigo do método adaptativo proposto. As fases desse algoritmo são, a seguir, detalhadas.

```
Algoritmo AGA(maxger, numind, probcrossover, p_m);
    t ← 0;
    Gere a população inicial P(t);
    Avalie P(t);
    enquanto (t ≤ maxger) faça
        t ← t + 1;
        Gere P(t) a partir de P(t-1);
        i ← 0;
        enquanto (i ≤ numind) faça
            cross ← Número aleatório de 1 a 100;
            se (cross ≤ probcrossover) faça
                Selecionar indivíduo;
                Selecionar Operador Crossover;
                Cruzar;
            fim-se
            i ← i + 1;
        fim-enquanto
        Avaliar P(t);
        Definir a população sobrevivente;
        Aplicar mutação com probabilidade p_m à população sobrevivente
```

```
            se (t_mod = 0) faça
                Atualizar probabilidade de escolha de cada operador (p(Oi));
                Executar Busca Local;
                Executar Reconexão por Caminhos;
            fim-se
        fim-enquanto
fim AGA
```

Figura 2 – Pseudocódigo do método adaptativo proposto

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.

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 E 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, seja O_i, com $i = 1, ..., 5$, 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 gerações, faz-se $A_i = 0$. Seja $q_i = f(s^*) * A_i$ e $p(O_i) = q_i / \sum_{j=1}^{5} q_j$ onde $i = 1, ... , 5$. Observe que, quanto melhor a solução, maior o valor de q_i e consequentemente, maior a probabilidade $p(Oi)$ 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 e Ribeiro (2000).

5.4.1 Busca Local

A busca local é aplicada periodicamente 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 funciona como se 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, 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.

5.4.2 Reconexão por Caminhos

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.

5.4.3 Sobrevivência de indivíduos

A sobrevivência de indivíduos da população determina quais deles permanecerão na população na próxima geração durante a evolução do algoritmo. No algoritmo proposto, sobrevivem os indivíduos mais adaptados, no caso, os integrantes do grupo elite, bem como os melhores indivíduos da população corrente, selecionados através da técnica do elitismo.

5.4.4 Mutação

Uma determinada quantidade de indivíduos, definida por uma probabilidade de mutação, é selecionada aleatoriamente da geração corrente e submetida à mutação por movimentos de troca entre duas tarefas. A aplicação de mutação nestes indivíduos contribui para a ocorrência de diversidade na população.

5.4.5 Critério de parada

Como critério de parada, adota-se o número máximo de gerações. No algoritmo implementado evolui-se até um máximo de 100 gerações.

5.4.6 Variantes do Algoritmo Genético Adaptativo (AGA)

Na variante AGA1, há atualização da probabilidade de seleção do operador *crossover* a cada cinco gerações. Após essa quantidade de gerações, todos os integrantes do grupo elite são submetidos a uma busca local e, em seguida, à Reconexão por Caminhos. Este algoritmo também diferencia-se dos demais por ter o grupo elite composto pelo melhor indivíduo produzido por cada operador *crossover* ao longo dessas cinco gerações.

A variante AGA 2 também promove a atualização do parâmetro de seleção do operador *crossover* a ser utilizado, a submissão do grupo elite à busca local e à Reconexão por Caminhos a cada cinco gerações. Ele se diferencia do AGA 1 pela composição do grupo elite, formado pelas cinco melhores soluções construídas pelo método globalmente e não apenas nas últimas cinco gerações.

Na variante AGA 3, a atualização do parâmetro de seleção do operador *crossover* a ser utilizado, a submissão do grupo elite à busca local e à Reconexão por Caminhos acontece a cada dez gerações. Já o grupo elite é composto pelas três melhores soluções produzidas pelo método até então, acrescidos da melhor solução produzida pelo método nas últimas dez gerações e desde que tenha índice de diversidade superior a 30% em relação às demais soluções já inseridas no grupo. Caso a melhor solução produzida ao longo das dez últimas gerações não se enquadre no critério apresentado, a segunda melhor solução é então analisada e caso também não se adeque aos critérios apresentados os testes serão realizados nas próximas soluções da sequência até que uma seja selecionada. O quinto indivíduo é escolhido a partir da seleção aleatória de um indivíduo do conjunto dos dez melhores indivíduos produzidos ao longo das dez gerações anteriores.

O índice de diversidade aplicado neste método é gerado a partir do somatório do número de

genes diferentes nas mesmas posições das duas soluções comparadas dividido pelo total de posições da solução.

6. Resultados Computacionais

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

Parâmetros	Valores
Parâmetro γ da fase de construção GRASP	0,20
Número máximo de iterações da busca local (*MRDMax*)	7 x *n*
Número máximo de gerações do método adaptativo proposto	100
Possibilidade de Cruzamento	80%
Taxa de mutação	5%

Tabela 1 – Parâmetros utilizados pelo método

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. Foram utilizados três conjuntos de problemas-teste: O primeiro é o mesmo utilizado por Gomes Jr. *et al.* (2007) e envolvem um número de tarefas *n* igual a 6 7, 8, 9, 10, 11, 12, 15, 20, 25, 30, 40, 50 e 75. O segundo e terceiro conjuntos foram propostos por Rosa et al. (2009). O segundo envolve um número de tarefas de *n* igual a 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19 e 20. O terceiro conjunto envolve um número de tarefas *n* igual a 6, 7, 8, 9, 10, 11, 12, 15, 20, 25, 30, 40, 50, 75 e 100.

Foram realizadas três baterias de testes, uma para cada conjunto de problemas. A primeira bateria de testes (BAT1) utilizou os problemas-teste propostos por Gomes Jr. et al. (2007). Cada problema-teste foi resolvido 30 vezes por cada uma das três variações implementadas do método proposto, denominados AGA 1, AGA 2 e AGA 3. Os problemas envolvendo 75 tarefas, foram resolvidos apenas 20 vezes, por demandarem um tempo computacional mais elevado. Além disso, para esses problemas também foi adotado o valor *MRDmax = 2 x n* como número máximo de iterações sem melhora da busca local para reduzir o tempo de processamento do algoritmo. A segunda bateria de testes (BAT2) foi realizada com os problemas-teste de Rosa et al. (2009). Cada problema-teste foi resolvido 30 vezes pelas três variantes implementadas: AGA 1, AGA 2 e AGA 3. A terceira e última bateria de testes (BAT 3) utilizou os problemas de Rosa et al. (2009). Cada problema-teste foi resolvido 30 vezes pelo método proposto, com exceção daqueles envolvendo 75 e 100 tarefas, os quais foram resolvidos apenas 10 vezes, por demandar um tempo computacional mais elevado. Para os problemas-teste de 75 e 100 tarefas adotou-se, também, o valor *MRDmax = 2 x n* como número máximo de iterações sem melhora da busca local para reduzir o tempo de processamento do algoritmo. Este conjunto de problemas-teste foi resolvido pelas três variantes implementadas AGA 1, AGA 2 e AGA 3.

As Tabelas 2, 3 e 4 mostram os resultados obtidos pelas baterias de teste BAT1, BAT2 e BAT3 para os algoritmos AGA 1, AGA 2 e AGA 3. A primeira coluna apresenta o número de tarefas envolvidas. Na segunda, terceira e quarta colunas é mostrado o quanto as médias das soluções de cada algoritmo (AGA 1, AGA 2 e AGA 3, respectivamente) desviaram da melhor solução conhecida referente a cada problema-teste. Já na quinta, na sexta e na sétima coluna é mostrado o quanto as melhores soluções geradas por tais algoritmos desviaram das melhores soluções conhecidas. O *Desvio sol. Média* é calculado pela expressão *Desvio sol. média = (RMed - MR)/MR*, sendo *RMed* o resultado médio obtido pela aplicação do respectivo algoritmo e *MR* o melhor resultado conhecido de cada problema-teste, obtidos por (Penna, 2009). Já para o *Desvio melhor solução*, a expressão *Desvio melhor solução =*

(RMelhor-MRl)/MRl, sendo *RMelhor* o melhor resultado obtido pela aplicação do respectivo algoritmo e *MRl* o melhor resultado conhecido de cada problema-teste, obtidos por (Penna, 2009). Na oitava, na nona e na décima coluna é exibido o tempo computacional médio de cada variante do algoritmo proposto.

Tarefas	Desvio sol. média			Desvio melhor solução			Tempo (s)		
	AGA 1	AGA 2	AGA 3	AGA 1	AGA 2	AGA 3	AGA 1	AGA 2	AGA 3
8	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,94	0,93	0,7
9	0,15%	0,16%	0,24%	0,00%	0,00%	0,00%	1,26	1,25	0,78
10	0,24%	0,25%	0,41%	0,00%	0,00%	0,00%	1,6	1,59	0,99
11	0,03%	0,05%	0,10%	0,00%	0,00%	0,00%	2,21	2,2	1,64
12	0,07%	0,08%	0,21%	0,00%	0,00%	0,00%	2,81	2,8	2,44
15	0,76%	0,80%	1,16%	0,00%	0,00%	0,00%	6,02	5,98	6,09
20	0,73%	0,75%	0,85%	0,00%	0,00%	0,00%	20,6	20,47	17,87
25	1,02%	1,08%	1,42%	0,00%	0,00%	0,00%	45,72	45,44	39,65
30	1,60%	1,82%	2,64%	0,00%	0,00%	0,00%	112,06	111,38	41,65
40	2,33%	2,34%	3,56%	0,08%	-0,08%	-0,09%	335,88	333,81	41,61
50	4,06%	4,37%	6,32%	0,02%	-0,31%	-1,11%	896,1	890,6	222,04
75	6,52%	9,48%	11,86%	0,04%	-4,40%	-1,76%	2000,05	1992,73	1242,06
Média	**1,46%**	**1,77%**	**2,40%**	**0,01%**	**-0,40%**	**-0,25%**	**285,85**	**284,1**	**134,79**

Tabela 2 – Resultados da primeira bateria de testes – BAT1

Tarefas	Desvio sol. média			Desvio melhor solução			Tempo (s)		
	AGA 1	AGA 2	AGA 3	AGA 1	AGA 2	AGA 3	AGA 1	AGA 2	AGA 3
6	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,65	0,71	0,65
7	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,86	1,33	0,82
8	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	1,13	1,22	1,01
9	0,00%	0,01%	0,00%	0,00%	0,01%	0,00%	1,47	1,76	1,26
10	0,00%	0,00%	0,03%	0,00%	0,00%	0,03%	1,78	2,21	1,53
11	0,00%	0,07%	0,12%	0,00%	0,07%	0,12%	2,37	2,81	1,91
12	0,00%	0,03%	0,04%	0,00%	0,03%	0,04%	3,88	3,8	2,44
13	0,00%	0,10%	0,20%	0,00%	0,10%	0,20%	4,87	5,37	3,09
14	0,00%	0,06%	0,14%	0,00%	0,06%	0,14%	5,75	6,14	3,47
15	0,02%	0,19%	0,27%	0,19%	0,19%	0,27%	8,5	7,99	4,43
16	0,06%	0,20%	0,20%	0,06%	0,20%	0,20%	11,64	8,97	2,69
17	0,00%	0,31%	0,31%	0,00%	0,31%	0,31%	0,61	10,88	3,26
18	0,00%	0,42%	0,42%	0,00%	0,42%	0,42%	0,61	15,23	4,57
19	0,04%	0,23%	0,23%	0,04%	0,23%	0,25%	18,17	19,09	5,73
20	0,02%	0,43%	0,44%	0,02%	0,43%	0,44%	30,32	17,63	10,95
Média	**0,01%**	**0,14%**	**0,16%**	**0,02%**	**0,14%**	**0,16%**	**6,17**	**7,01**	**3,19**

Tabela 3 – Resultados da primeira bateria de testes – BAT2

Tarefas	Desvio sol. média			Desvio melhor solução			Tempo (s)		
	AGA 1	AGA 2	AGA 3	AGA 1	AGA 2	AGA 3	AGA 1	AGA 2	AGA 3
6	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,76	0,61	0,65
7	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,76	0,94	0,81
8	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	1,03	1,21	1
9	0,00%	0,13%	0,20%	0,00%	0,13%	0,20%	1,35	1,53	1,26
10	0,00%	0,03%	0,06%	0,00%	0,03%	0,06%	1,61	1,88	1,65
11	0,00%	0,11%	0,17%	5,97%	0,11%	6,16%	2,2	0,73	1,92
12	0,00%	0,01%	0,11%	0,00%	0,01%	0,11%	3,32	3,09	2,37
15	0,01%	0,14%	0,37%	0,01%	0,14%	0,37%	7,49	5,6	4,38
20	0,30%	0,64%	0,81%	0,30%	0,64%	0,81%	23,88	10,18	10,61
30	1,20%	1,19%	1,35%	1,36%	1,51%	1,50%	172,51	64,19	47,13
40	0,98%	1,21%	0,98%	1,03%	1,42%	2,34%	801,67	155,05	98,05
50	1,14%	1,46%	1,14%	1,26%	2,24%	2,58%	1575,11	529,11	416,78
75	0,00%	1,26%	2,36%	0,09%	1,67%	3,60%	4978,02	3381,58	1398,82
100	0,25%	2,50%	1,10%	0,25%	3,14%	2,41%	18107,72	23209,42	15853,97
Média	0,28%	0,62%	0,62%	0,73%	0,79%	1,44%	1834,1	1954,82	1274,24

Tabela 4 – Resultados da primeira bateria de testes – BAT3

Na Tabela 5 são comparados os resultados obtidos pelo AGA 1 em BAT1 e aqueles alcançados pelo algoritmo GTSPR, proposto por Penna (2009). 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 1 e GTSPR, respectivamente) desviaram da melhor solução conhecida referente a cada problema-teste. Na quarta coluna é apresentado o percentual de melhora para o valor médio obtido por comparação entre os dois métodos. Na quinta e na sexta coluna é mostrado o quanto as melhores soluções geradas por tais algoritmos desviaram das melhores soluções conhecidas. Na sétima coluna o percentual de melhora referente ao melhor valor conhecido é exibido. Em seguida, nas duas últimas colunas, são exibidos os tempos computacionais médios requeridos por cada algoritmo.

Tarefas	Desvio sol. média			Desvio melhor solução			Tempo (s)	
	AGA 1	GTPRS	% Melhora	AGA 1	GTPRS	% Melhora	AGA 1	GTPRS
8	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,94	0,06
9	0,15%	0,00%	-15,00%	0,00%	0,00%	0,00%	1,26	0,09
10	0,24%	0,00%	-24,00%	0,00%	0,00%	0,00%	1,6	0,15
11	0,03%	0,00%	-3,00%	0,00%	0,00%	0,00%	2,21	0,25
12	0,07%	0,00%	-7,00%	0,00%	0,00%	0,00%	2,81	0,36
15	0,76%	1,25%	64,08%	0,00%	0,00%	0,00%	6,02	0,46
20	0,73%	1,11%	51,87%	0,00%	0,00%	0,00%	20,6	2,05
25	1,02%	1,60%	56,53%	0,00%	0,00%	0,00%	45,72	6,62
30	1,60%	2,57%	60,61%	0,00%	0,00%	0,00%	112,06	18,66
40	2,33%	3,77%	61,84%	0,08%	0,00%	8,00%	335,88	84,16
50	4,06%	5,58%	37,64%	0,02%	0,00%	2,00%	896,1	305,28
75	6,52%	7,41%	13,69%	0,04%	0,00%	4,00%	2005,05	3.472,26
Média	**1,46%**	**2,01%**	**2,27%**	**0,01%**	**-0,40%**	**-0,25%**	**285,85**	**324,2**

Tabela 5 – Comparação BAT1: AGA1 x GTPRS

A tabela 6 exibe a comparação dos resultados da variante AGA1 em BAT2 e a formulação de programação matemática MPLIM-IT, de Rosa et al.(2009), que representa o único resultado encontrado na literatura para este conjunto de problemas.

Tarefas	Desvio sol. média		Desvio melhor solução		Tempo (s)	
	AGA 1	MPLIM-IT	AGA 1	MPLIM-IT	AGA 1	MPLIM-IT
6	0,00%	0,00%	0,00%	0,00%	0,65	0,71
7	0,00%	0,00%	0,00%	0,00%	0,86	1,33
8	0,00%	0,00%	0,00%	0,00%	1,13	1,22
9	0,00%	0,01%	0,00%	0,01%	1,47	1,76
10	0,00%	0,00%	0,00%	0,00%	1,78	2,21
11	0,00%	0,07%	0,00%	0,07%	2,37	2,81
12	0,00%	0,03%	0,00%	0,03%	3,88	3,8
13	0,00%	0,10%	0,00%	0,10%	4,87	5,37
14	0,00%	0,06%	0,00%	0,06%	5,75	6,14
15	0,02%	0,19%	0,19%	0,19%	8,5	7,99
16	0,06%	0,20%	0,06%	0,20%	11,64	8,97
17	0,00%	0,31%	0,00%	0,31%	0,61	10,88
18	0,00%	0,42%	0,00%	0,42%	0,61	15,23
19	0,04%	0,23%	0,04%	0,23%	18,17	19,09
20	0,02%	0,43%	0,02%	0,43%	30,32	17,63
Média	**0,01%**	**0,14%**	**0,02%**	**0,14%**	**6,17**	**7,01**

Tabela 6 – Comparativo BAT2: AGA1 x MPLIM-IT

7. Conclusões

Este trabalho teve seu foco no Problema de Sequenciamento em uma máquina com penalidades por antecipação e atraso da produção, considerando janelas de entrega e tempo de preparação da máquina dependente da sequência de produção. Para resolvê-lo foi proposto um Algoritmo Genético Adaptativo. A população inicial deste algoritmo é gerada por um procedimento GRASP, utilizando como função guia as regras de despacho.

Durante o processo evolutivo, a população passa pelas fases de seleção, cruzamento e mutação. No cruzamento, cinco operadores *crossover* são utilizados, sendo que a escolha de qual operador será empregado depende da qualidade das soluções produzidas em gerações passadas por estes operadores. Nesse período, é criado um grupo elite, formado por indivíduos de qualidade. Periodicamente, esses indivíduos do grupo elite são submetidos a uma busca local e ao procedimento Reconexão por Caminhos. O procedimento de Reconexão por Caminhos liga a melhor solução produzida até então a cada uma das soluções do grupo elite.

Foram desenvolvidas três variantes do algoritmo, denominadas AGA 1, AGA2 e AGA 3. Essas variantes diferem entre si pela composição dos indivíduos do grupo elite. No AGA 1, o grupo elite é formado pelas melhores soluções geradas nas últimas 5 gerações por cada um dos operadores; enquanto no AGA 2, esse grupo contém os melhores indivíduos formados até então por cada operador. Já no AGA 3, o grupo elite mescla melhores indivíduos formados nas últimas 10 gerações com outros que satisfazem a um determinado índice de diversidade. Para testar estas variantes, foram realizadas três baterias de testes, cada uma delas usando um conjunto de problemas-teste que se diferenciavam basicamente na forma de geração dos tempos de preparação da máquina.

Na primeira bateria, as três variantes foram submetidas aos problemas-teste Gomes Jr. et al. (2007). A variante AGA 1, que obteve o melhor desempenho, teve os seus resultados comparados com os Penna (2009), que havia se mostrado superior ao de Gomes Jr. et al. (2007). A variante AGA 1 se mostrou superior por apresentar menor variabilidade das soluções finais e menores tempos computacionais. Na segunda bateria, foram utilizados problemas-testes de até 20 tarefas, propostos por Rosa et al. (2009). A variante AGA 1 também apresentou o melhor desempenho. Os resultados também foram comparados com

aqueles alcançados por Rosa et al. (2009), e verificou-se que o AGA 1 foi capaz de gerar todas as soluções ótimas, sem nenhuma variabilidade nas soluções finais, em tempos computacionais muito menores que os desses autores. A terceira bateria de testes utilizou problemas-teste envolvendo até 100 tarefas, criados a partir da proposta de Rosa et al. (2009). A variante AGA 1 se destacou por apresentar menor variabilidade; mas, por outro lado, um alto custo computacional. Não houve comparação com outros trabalhos da literatura esse conjunto de problemas-teste ser novo. Os tempos computacionais obtidos pelas variantes do algoritmo heurístico proposto foram razoavelmente pequenos se comparados ao horizonte de planejamento, que, tipicamente, é de uma semana. Os problemas de pequenas dimensões (6 até 20 tarefas) foram resolvidos rapidamente em todos os conjuntos de problemas-teste. Os problemas de médio porte (30 a 50 tarefas) requereram um tempo maior, de até 1,5 horas. Já os problemas maiores exigiram um esforço computacional bem maior, de aproximadamente 21 horas de processamento e um esforço no sentido de diminuir tais tempos deve ser feito.

Agradecimentos

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

Referências

ALLAHVERDI A., GUPTA J. N. D. & ALDOWAISAN T. *(1999), A review of scheduling research involving setup considerations, OMEGA, v. 27, p. 219-239.*

BARCELLOS, J. C. HOLLAND DE. *(2000). Algoritmos genéticos adaptativos: Um estudo comparativo. Dissertação de mestrado, Escola Politécnica da USP, São Paulo.*

BUSTAMANTE, L. M.. *(2006), Minimização do Custo de Antecipação e Atraso para o Problema de Sequenciamento de uma Máquina com Tempo de Preparação Dependente da Sequência: Aplicação em uma Usina Siderúrgica. Dissertação de mestrado, Programa de Pós-Graduação em Engenharia de Produção, UFMG, Belo Horizonte.*

DU, J. & LEUNG, J. Y. T. *(1990), Minimizing total tardiness on one machine is NP-hard, Mathematics of Operations Research, v. 15, p. 483-495.*

FELDMANN, M. & BISKUP, D. *(2003), Single-Machine Scheduling for Minimizing Earliness and Tardiness Penalties by Meta-heuristic Approaches, Computers and Industrial Engineering, v. 44, p. 307–323.*

FEO, T. A. & RESENDE, M. G. C. *(1995), Greedy randomized adaptive search procedures, Journal of Global Optimization, v. 6, p. 109–133.*

FRANÇA FILHO, M. F. *(2007), GRASP e Busca Tabu aplicados a problemas de programação de tarefas em máquinas paralelas, Tese de doutorado, Campinas: Departamento de Engenharia de Sistemas, UNICAMP.*

GLOVER, F. *(1986), Future paths for Integer Programming and links to Artificial Intelligence, Computers and Operations Research, v. 5, p. 553–549.*

GOMES JR., A. C.; CARVALHO, C. R. V.; MUNHOZ, P. L. A. & SOUZA, M. J. F. *(2007), Um método heurístico híbrido para a resolução do problema de sequenciamento em uma máquina com penalidades por antecipação e atraso da produção, In Anais do XXXIX Simpósio Brasileiro de Pesquisa Operacional, SBPO, Fortaleza, p. 1649–1660.*

HINO, C. M.; RONCONI, D. P. & MENDES, A. B. *(2005), Minimizing Earliness and Tardiness Penalties in a Single-Machine Problem with a Common Due Date, European Journal of Operational Research, v. 160, p. 190–201.*

LEE, C. Y. & 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, v. 22, 857–869.*

MANNE, A. S. *(1960), On the Job-shop Scheduling Problem, Operations Research, v. 8, p. 219–223*

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

PENNA, P. H. V. *(2009). Um algoritmo heurístico híbrido para minimizar os custos com a antecipação e o atraso da produção em ambientes com janelas de entrega e tempos de preparação dependentes da sequência de produção. Dissertação de mestrado, Programa de Pós-Graduação em Engenharia Mineral, Escola de Minas, UFOP, Ouro Preto.*

PRAIS, M. & RIBEIRO, C. C. *(2000). An application to a matrix decomposition problem in tdma traffic assignment. INFORMS - Journal on Computing, v. 12, p. 164_176.*

ROSA, B. F.; SOUZA, M. J. F. & SOUZA, S. R. *(2009). Formulações de programação matemática para o problema de sequenciamento em uma máquina com janelas de entrega distintas e tempo de preparação dependentes da sequência de produção. Anais do XXXII Congresso Nacional de Matemática Aplicada e Computacional - CNMAC 2009, Cuiabá.*

ROSSETTI, I. C. M. *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, (2003).*

RUBIN, P. A. & RAGATZ, G. L. *(1995). Scheduling in a sequence dependent setup environment with genetic search. Computers and Operations Research, v. 22, p. 85_89.*

SOUZA, M. J. F. *(2008). Inteligência Computacional para Otimização. Notas de aula, Departamento de Computação, UFOP, Ouro Preto-MG.*

WAN, G. & 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, v. 142, p. 271–281.*

YING, K. C. *(2008), Minimizing earliness–tardiness penalties for common due date single-machine scheduling problems by a recovering beam search algorithm, Computers and Industrial Engineering, v. 55, p. 494-502.*