SPOLM 2009

ISSN 2175-6295 Rio de Janeiro- Brasil, 05 e 06 de agosto de 2009.

# 078/2009 - TÉCNICAS HEURÍSTICAS EVOLUTIVAS ADAPTATIVAS APLICADAS AO PROBLEMA DE SEQUENCIAMENTO EM UMA MÁQUINA COM PENALIZAÇÃO POR ANTECIPAÇÃO E ATRASO DA PRODUÇÃO

**Fábio Fernandes Ribeiro**
Centro Federal de Educação Tecnológica de Minas Gerais
Avenida Amazonas 7675 - Nova Gameleira – 30.510-000 – Belo Horizonte – MG - Brasil
fabiobh@gmail.com

**Marcone Jamilson Freitas Souza**
Universidade Federal de Ouro Preto
ICEB, Campus Universitário – Morro do Cruzeiro – 30.510-000 – Ouro Preto – MG - Brasil
marcone@iceb.ufop.br

**Sérgio Ricardo de Souza**
Centro Federal de Educação Tecnológica de Minas Gerais
Avenida Amazonas 7675 - Nova Gameleira – 30.510-000 – Belo Horizonte – MG - Brasil
sergio@dppg.cefetmg.br

**Resumo**

Este trabalho aborda o problema de sequenciamento em uma máquina com penalização por antecipação e atraso da produção, considerando janelas de entrega e tempo de preparação de máquina dependente da sequência de produção. Devido à complexidade combinatória do problema, classificado como NP-difícil, propõe-se resolvê-lo a partir da aplicação de técnicas heurísticas evolutivas, baseada em algoritmos genéticos, e adaptativa baseada em GRASP Reativo. 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 apresentam a efetividade do algoritmo proposto.

**Palavras-Chaves:** Algoritmos Genéticos; GRASP; Metaheurísticas; Reconexão por caminhos; Sequenciamento em uma máquina.

**Abstract**

This work deals the total tardiness single machine scheduling problem with earliness and tardiness penalties, considering due dates and sequence-dependent setup time. According to its complexity, classified by NP-hard, rolling heuristics techniques, based on genetic

algorithm technique, and adaptive technique, based on Reactive GRASP, are applied to solve it. For each individual (job sequence) generated a polynomial time algorithm are used to determine optimal initial processing date to each job in a sequence given. Five search operators are used to explore the solutions space where the choice probability for each operator depends to success in a previous search. Computational results show the effectiveness of the proposed algorithm.
**Keywords:** Genetic Algorithm; GRASP; Metaheuristics; Path Relinking; Single Machine Scheduling;

## 1. INTRODUÇÃO

Os problemas de sequenciamento estão entre os mais estudados na atualidade. Em [1] ressalta-se a importância desta classe de problemas, visto o grande número de aplicações práticas na indústria. [3], 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.

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 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.

Uma metodologia evolutiva adaptativa, baseada em Algoritmos Genéticos e GRASP Reativo é proposta para a resolução deste problema. Inicialmente, constrói-se uma população inicial a partir da meta heurística GRASP, proposta por [6], tendo como função gulosa cinco regras de despacho (TDD, EDD, SPT, WSPT e LPT). Após a construção da população inicial, dois indivíduos (sequência de tarefas) são selecionados para cruzamento, de acordo com a probabilidade de cruzamento estabelecida, onde um método adaptativo inspirado em GRASP Reativo [16] é utilizado para a escolha do operador *crossover* a ser empregado. Esse método é baseado na qualidade dos indivíduos gerados por cada operador *crossover* ao longo da evolução, ou seja, após certo número de gerações, 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. Neste trabalho são utilizadas também duas metodologias para a criação de um conjunto de soluções “elite”, que será utilizado durante o processo evolutivo. O primeiro conjunto elite, denominado “CROSS”, conterá a melhor solução produzida por cada operador *crossover* num intervalo de 5 gerações. Já o segundo conjunto elite, denominado “BEST”, será composto pelas 5 melhores soluções produzidas pelo método evolutivo adaptativo até então.

A população sobrevivente é composta pelos cinco indivíduos do grupo elite CROSS, pelos cinco indivíduos do conjunto elite BEST, pelo melhor indivíduo encontrado até então e por 85 indivíduos selecionados através da técnica do elitismo. Outros cinco indivíduos são escolhidos aleatoriamente na geração corrente e submetidos à mutação, onde um movimento de realocação de tarefas é aplicado, com o objetivo de assegurar a diversidade da população.

A população evolui até que o critério de parada seja atingido. Por último, a Reconexão por Caminhos [9] é também aplicada a cada cinco gerações, e foi implementada caminhando-se do melhor indivíduo produzido pelo algoritmo evolutivo adaptativo até então, em direção a cada um dos indivíduos do grupo elite.

O restante deste trabalho está estruturado como se segue. Na Seção “Trabalhos Relacionados” é feita uma apresentação de trabalhos relacionados ao problema. As características do problema estudado são detalhadas na Seção “Problema Abordado” e na seção “Modelagem Matemática”, o modelo matemático para o problema abordado, é apresentado. Já na Seção “Metodologia Heurística”, a metodologia proposta para resolução do PSUMAA, é descrita. Na Seção “Resultados Computacionais” são apresentados e discutidos os resultados computacionais. A Seção “Conclusões” conclui o trabalho e aponta perspectivas futuras para melhoramento do método 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 [4].

Em [22] é resolvido o PSUMAA 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.

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 [7]. 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.

Em [17] é estudado o PSUMAA com datas comuns de entrega e tempo de preparação da máquina dependente da sequência de produção. Os autores apresentaram um procedimento *branch-and-bound* que representou um considerável avanço nos estudos desta classe de problemas até aquela época, por ser capaz de resolver na otimalidade, com um esforço computacional aceitável, problemas-teste de até 25 tarefas.

No trabalho de [10], 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 [21], 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 [10] é propoto um algoritmo baseado em GRASP, Descida em Vizinhança Variável (VND), Busca Tabu e Reconexão por Caminhos para resolver o PSUMAA com janelas de entrega e tempo de preparação dependente da sequência de produção. Uma solução inicial é gerada pelo método GRASP e em seguida, submetida a um refinamento através da heurística de Descida em Vizinhança Variável (VND). A partir desta solução, o método de Busca Tabu é então aplicado até que um critério de parada seja atingido. A Reconexão por Caminhos é então utilizada como ferramenta de pós-refinamento e é aplicada após a execução da BT.

Em [15] é desenvolvido um algoritmo em três fases combinando os procedimentos GRASP, VND, Busca Tabu e Reconexão por caminhos. Na primeira fase os autores utilizaram um procedimento GRASP para gerar a população inicial, que são refinadas em seguida pelo método VND, que utiliza movimentos de realocação e troca de tarefas. Na segunda fase aplica-se Busca Tabu à solução inicial refinada e na terceira fase a Reconexão por caminhos e então aplicada a pares de soluções do grupo elite.

Em [12] os autores utilizaram Algoritmos Genéticos (AG) para a resolução do PSUMAA com datas de entrega distintas. Nesse 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.

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 [2].

Segundo [14], 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 Sij 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 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 $\alpha i$ e um custo unitário de atraso $\beta i$. 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 [10], que se baseou nos trabalhos de [3] e [13].

Sejam si, 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 [3] foram utilizados 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 $P_0$ 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-se ainda, uma constante M de valor suficientemente grande. O modelo de PLIM proposto em [10] é apresentado a seguir:

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

sujeito a:

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

$$\sum_{j=1,\, j\neq i}^{n+1} y_{ij} = 1 \quad \forall\ j = 1, 2, ..., n+1 \text{ e } i \neq j \tag{3.3}$$

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

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

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

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

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

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

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

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, elas 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. 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 (3.1) do modelo de PLIM, em que são considerados mais adaptados aqueles indivíduos que obtiverem o menor valor.

## 5.3. CONSTRUÇÃO DA POPULAÇÃO INICIAL

A população inicial do método adaptativo proposto é gerada aplicando-se a fase de construção GRASP [6] tendo, como função guia, cinco regras de despacho (EDD, TDD, SPT, WSPT e LPT), descritas na seção 4.3.1.

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*) constrói 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 atendimento. 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 primeiras a serem processadas.

### 5.3.2. Fase de construção GRASP

O procedimento de construção segue as ideias da fase de construção do algoritmo GRASP [6]. 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-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.

A Figura 1 apresenta o pseudocódigo da fase de construção. Nesta figura, $g_{min}$ representa o melhor valor assumido pela Regra de Despacho escolhida e $g_{max}$, o maior.

**procedimento** *Construcao (g(.), γ, 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
**fim** *Construcao;*

Figura 1 – Procedimento para construção de indivíduos

### 5.4. MÉTODO EVOLUTIVO ADAPTATIVO APLICADO AO PSUMAA

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

**Algoritmo** *MEA(maxger, numind, probcrossover, probmutacao);*
*t* ← *0;*
Gere a população inicial *P(t)*;
Avalie *P(t);*
enquanto (*t*≤ *maxger)* faça
$t \leftarrow t + 1$;
Gere *P(t)* a partir de *P(t-1);*
enquanto (i ≤ *numind)* faça
$i \leftarrow 1$;
$i \leftarrow$ Número aleatório de 1 a 100;
se (*cross* ≤ *probcrossover*) faça
Selecionar indivíduos;
Cruzar
fim-se
Avaliar *P(t);*
fim-enquanto
Definir a população sobrevivente;

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

Figura 2 – Pseudocódigo do método evolutivo adaptativo proposto

#### 5.4.1. 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.

#### 5.4.2. 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 [12].

O operador *One Point Crossover* (OX) seleciona, randomicamente, um ponto de corte. As tarefas à direita do ponto de corte do Pai 1 e 2 são copiadas para os filhos 1 e 2, respectivamente. Os filhos 1 e 2 são, então, completados, seguindo a sequência de tarefas dos pais 2 e 1, respectivamente.

No operador Similar *Job Order Crossover* (SJOX), os dois pais são examinados, tarefa por tarefa. Caso uma mesma tarefa apareça em uma mesma posição em ambos os pais, ela é copiada para os dois filhos. Na sequência, um ponto de corte é escolhido randomicamente e as tarefas faltantes nos filhos 1 e 2 são copiadas dos pais 1 e 2, respectivamente. Por último, as tarefas à esquerda do ponto de corte são completadas, seguindo a sequência de tarefas dos pais 2 e 1 para os filhos 1 e 2, respectivamente.

Proposto por [19] especificamente para problemas de sequenciamento, o operador *Relative Job Order Crossover* (RRX) não apenas preserva a ordem das tarefas dos pais mas também, preserva algumas tarefas em suas posições absolutas na sequência. Ele divide as tarefas em dois conjuntos e as mistura de acordo com a ordem de ambos os pais, sem decisões aleatórias. De acordo com este esquema, oito indivíduos são gerados, sendo dois deles clones dos pais. O operador RRX implementado neste trabalho não utiliza os dois clones gerados. Como um tamanho fixo de população é adotado, torna-se necessária a seleção de alguns dos indivíduos gerados para incorporá-los à nova população. Neste trabalho, dos seis indivíduos gerados, apenas aqueles com menor valor da função de avaliação são aproveitados. O processo de cruzamento dos indivíduos divide-se em duas fases. Na primeira fase, os pais são divididos em duas partes a partir da escolha de um ponto aleatório e as primeiras partes de cada indivíduo são copiadas, de um para outro. A segunda fase do cruzamento consiste em acrescentar, aos indivíduos faltantes, os genes, na sequência do pai que contribuirá com o segundo fragmento. Por exemplo, se um filho recebeu genes do pai 1 na primeira fase, na segunda fase ele receberá os genes do pai 2 e assim sucessivamente.

O operador *Based Order Uniform Crossover* (BOUX) foi desenvolvido com o objetivo de evitar que soluções inviáveis sejam construídas. Segundo [12], o BOUX é considerado um dos operadores que melhor se adéqua ao PSUMAA. O procedimento de cruzamento se inicia com a construção de uma string com o mesmo tamanho dos pais, que

armazenará valores 0 ou 1, escolhidos aleatoriamente. Em seguida, o cruzamento é realizado gene a gene, respeitando as seguintes premissas: caso o bit seja 0, o filho 1 recebe o gene da mesma posição do pai 1 e o filho 2 do pai 2; caso o bit seja 1, o filho 1 recebe o gene da mesma posição do pai 2 e o filho 2 do pai 1. Antes de um gene ser inserido na sequência, o procedimento verifica se o gene já consta no filho que está sendo gerado e, em caso positivo, o gene a ser inserido é o da mesma posição do outro pai. Se, ainda assim, o gene já estiver no filho é utilizada a sequência do pai indicado pelo bit.

O operador *Partially Mapped Crossover* (PMX) executa o mapeamento de um fragmento do pai no seu descendente. Os genes restantes são obtidos a partir da sequência do outro pai. No operador implementado são escolhidos aleatoriamente dois pontos de corte. O ponto de corte 1 sempre será menor que o ponto de corte 2. A escolha destes pontos possibilita a extração de um fragmento de cada pai. O filho 1 herda o fragmento do pai 2 e o filho 2, o fragmento do pai 1. Para manter a factibilidade dos indivíduos criados, os genes restantes são preenchidos seguindo a ordem do pai que não contribuiu com o fragmento, sempre observando se o gene a ser inserido já consta no filho.

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* Oi 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 $p(O_i) = q_i \Big/ \sum_{j=1}^{5} q_j$ , com $i = 1, ..., 5$. 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 em [16].

#### 5.4.3. 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 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.

#### 5.4.4. Reconexão por Caminhos

A Reconexão por Caminhos (*Path Relinking*) é um procedimento que integra estratégias de intensificação e diversificação, durante o processo de busca [18]. 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 conjunto de cinco soluções que serão utilizadas como função guia do procedimento de Reconexão por caminhos. Para a criação deste grupo, denominado conjunto elite, foram utilizados dois

procedimentos distintos, a saber:

i. Grupo formado com os cinco melhores indivíduos obtidos por cada operador *crossover*. Periodicamente, a cada 5 gerações, este grupo é então zerado e novos indivíduos são adicionados. Esta metodologia dá origem ao conjunto elite denominado "CROSS";
ii. Grupo formado pelos 5 melhores indivíduos produzidos pelo método evolutivo adaptativo até então. Esta metodologia dá origem ao conjunto elite chamado "BEST";

A cada 5 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 indivíduos constantes no conjunto elite BEST. 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.

### 5.4.5. Sobrevivência de indivíduos

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

### 5.4.6. 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.

## 6. RESULTADOS COMPUTACIONAIS

O método evolutivo 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 utilizados pelo método

| 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 |
| 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 [15] e envolvem um número de tarefas *n* igual a 8, 9, 10, 11, 12, 15, 20, 25, 30, 40, 50 e 75. São resolvidos 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 15 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. Esses ajustes nos parâmetros também foram adotados por [15].

A Tabela 2 mostra os resultados obtidos pelo método proposto, bem como reproduz os de [15]. A primeira coluna mostra o número de tarefas envolvidas em cada problema-teste. Na segunda e terceira colunas é mostrado o quanto as soluções médias de cada algoritmo (Método Evolutivo Adaptativo – MEA e Penna *et al*. (2008) [15], 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. O *Desvio* é calculado pela expressão *Desvio* = (*RMed* – *MR*) / *MR*, sendo *RMed* o resultado médio obtido pela aplicação do respectivo algoritmo e *MR* o melhor resultado conhecido de cada problema-teste. Em seguida, nas duas últimas colunas, são exibidos os tempos computacionais médios obtidos por cada metodologia.

Tabela 2 – Comparação de resultados entre os algoritmos MEA e Penna (2008) *et al*

| **Tarefas** | **Valor Médio** | | | **Melhor Valor** | | | **Tempo Médio** | |
|---|---|---|---|---|---|---|---|---|
| | **MEA** | **Penna (2008) *et al*** | **% Melhora** | **MEA** | **Penna (2008) *et al*** | **% Melhora** | **MEA** | **Penna (2008) *et al*** |
| 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 | 1,60 | 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 | 20,60 | 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 | 60,06 | 18,66 |
| 40 | 2,33 | 3,77 | 61,84 | 0,08 | 0,00 | 8,00 | 85,88 | 84,16 |
| 50 | 4,06 | 5,58 | 37,64 | 0,02 | 0,00 | 2,00 | 296,10 | 305,28 |
| 75 | 6,52 | 7,41 | 13,69 | 0,04 | 0,00 | 4,00 | 2005,05 | 3.472,26 |
| **Média** | 1,46 | 1,94 | 24,82 | 0,14 | 0,00 | 1,17 | 210,69 | 324,20 |

Conforme pode ser observado na Tabela 2, os valores médios obtidos pelo MEA apresentaram uma grande melhoria em comparação aos de [15]. A metodologia MEA atingiu valores médios menores do que os de [15] na maioria dos problemas-teste resolvidos. Este fato mostra a robustez do método adaptativo proposto, uma vez que produz soluções finais com uma menor variabilidade. Para os problemas-teste envolvendo 40, 50 e 75 tarefas, o método proposto foi capaz de produzir as melhores soluções para este problema conhecidas até então. Como desempenho global dos métodos, o método proposto por [15] apresentou um somatório dos desvios em relação à solução média de 23,29, ao passo que o método adaptativo atingiu 17,50 neste somatório. Em relação à melhoria da melhor solução encontrada para o problema, o método adaptativo apresentou um somatório de 1,17% enquanto o método proposto por [15], uma piora de 0,43%. Finalmente o tempo computacional médio obtido pelo método proposto foi de 210,69 s e o de [15] de 324,20 s.

## 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 método adaptativo baseado em algoritmos genéticos, onde a população inicial foi 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. Periodicamente, as soluções do grupo elite "BEST" são submetidas à busca local e à Reconexão por Caminhos. O procedimento de Reconexão por Caminhos liga a melhor solução produzida até então a cada uma das soluções deste grupo elite.

Para testá-lo, foram utilizados problemas-teste descritos na seção 6, sendo o mesmo comparado com um algoritmo da literatura. Para problemas envolvendo até 30 tarefas, o método proposto apresentou soluções de alta qualidade e com baixo desvio, sempre atingindo a melhor solução conhecida até então. 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.

## 8. REFERÊNCIAS BIBLIOGRÁFICAS


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

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

[3] 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.

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

[5] FELDMANN, M. E 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.

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

[7] 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.

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

[9] GLOVER, F. (1996), Tabu search and adaptive memory programming – Advances, applications and challenges, In Barr RS, Helgason RV, Kennington JL (Eds), *Computing Tools for Modeling, Optimization and Simulation: Interfaces in Computer Science and Operations Research*. Kluwer Academic Publishers, p. 1-75.

[10] GOMES JR., A. C.; CARVALHO, C. R. V.; MUNHOZ, P. L. A. E 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.

[11] HINO, C. M.; RONCONI, D. P. E 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.

[12] LEE, C. Y. E 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.

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

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

[15] PENNA, P.H.V.; SOUZA, M.J.F.; GONÇALVES, F.A.C.A E OCHI, L.S (2008), Uma heurística híbrida para minimizar custos com antecipação e atraso em sistemas de produção com janelas de entrega e tempos de preparação dependentes da sequência. SPOLM 2008, ISSN 1806-3632, Rio de Janeiro, Brasil.

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

[17] RABADI, G., MOLLAGHASEMI, M. E ANAGNOSTOPOULOS, G. C. (2004), A Branch-and-Bound Algorithm for the Early/Tardy Machine Scheduling Problem with a Common due-date and Sequence-Dependent Setup Time, *Computers & Operations Research*, 31, 1727–1751.

[18] 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.

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

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

[21] WAN, G. E 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.

[22] 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.