# ALGORITMO MULTIOBJETIVO DE BUSCA ADAPTATIVA EM VIZINHANÇA DE LARGA ESCALA PARA O SEQUENCIAMENTO DE TAREFAS EM MÁQUINAS CONSIDERANDO CONSUMO DE ENERGIA ELÉTRICA

LUCIANO PERDIGÃO COTA* FREDERICO GADELHA GUIMARÃES† FERNANDO BERNARDES DE OLIVEIRA‡
MARCONE JAMILSON FREITAS SOUZA§ GUILHERME MURTA MIRANDA§

*Instituto Tecnológico Vale
Av. Juscelino Kubitschek 31, 35400-000, Ouro Preto, MG, Brasil

†Laboratório MINDS, Departamento de Engenharia Elétrica - Universidade Federal de Minas Gerais
Av. Antônio Carlos 6627, 31270-901, Belo Horizonte, MG, Brasil

‡Departamento de Computação e Sistemas - Universidade Federal de Ouro Preto
Rua 36 115, 35931-008, João Monlevade, MG, Brasil

§Departamento de Computação - Universidade Federal de Ouro Preto
Campus Universitário Morro do Cruzeiro, 35400-000, Ouro Preto, MG, Brasil

Email: lucianocota@ufmg.br, fredericoguimaraes@ufmg.br,
fernando@decea.ufop.br, marcone@ufop.edu.br,
miranda.m.guilherme@gmail.com

**Abstract—** Industries are responsible for much of the world's energy consumption and pollutant emissions. There is a growing awareness of society on these issues and the emergence of increasingly rigid laws. Thereby, industries seek ways to improve their industrial processes. This work addresses the unrelated parallel machine scheduling problem with setup times, seeking to minimize the makespan and the total energy consumption. This version of the problem that minimizes the total energy consumption has been recently introduced in the literature and until then only exact methods were employed for its resolution. In this work, we propose an adaptive heuristic algorithm that uses learning techniques to improve the search process. In the computational experiments, we use instances from the literature and the results of an exact method for validation. The results were examined by the hypervolume indicator and graphical analysis. In this experiment environment, the results suggest that the proposed algorithm obtained a good convergence to the Pareto front and showed that it has practical applicability.

**Keywords—** Combinatorial optmization, machine scheduling, total energy consumption, adaptive large neighborhood search, learning automata

**Resumo—** As indústrias são responsáveis por grande parte do consumo de energia e emissão de poluentes no mundo. Com a crescente conscientização da sociedade por estes temas e o surgimento de legislações cada vez mais rígidas, as indústrias têm buscado maneiras de aperfeiçoar os seus processos industriais. Este trabalho trata o problema de sequenciamento em máquinas paralelas não relacionadas com tempos de preparação dependentes da sequência, buscando minimizar o *makespan* e o consumo total de energia. Esta versão do problema minimizando o consumo total de energia foi introduzida recentemente na literatura e até então foram aplicados apenas métodos exatos para a sua resolução. Neste trabalho é proposto um algoritmo heurístico adaptativo que utiliza técnicas de aprendizagem para aperfeiçoar o processo de busca. Nos experimentos computacionais são utilizadas instâncias da literatura e os resultados de um método exato. Os resultados foram examinados usando a métrica hipervolume e análise gráfica. Neste ambiente de experimentos, os resultados sugerem que o algoritmo proposto obteve uma boa convergência para a fronteira Pareto e mostrou que tem aplicabilidade prática.

**Palavras-chave—** Otimização combinatória, sequenciamento de máquinas, consumo total de energia, busca adaptativa em vizinhança de larga escala, aprendizagem com autômatos.

## 1 Introdução

Com o mercado global há uma competição acirrada por produtos de melhor qualidade e de menor custo. Assim, as indústrias precisam tornar seus processos industriais mais eficientes. Com isso, têm-se a necessidade de melhorar o planejamento de produção. Os problemas de sequenciamento em máquinas têm grande importância neste processo. Estes problemas envolvem a maximização da produção pela alocação de tarefas aos recursos disponíveis de maneira ótima.

Este trabalho trata especificamente do problema de sequenciamento em máquinas paralelas não relacionadas com tempos de preparação dependentes da sequência. Este problema tem grande importância prática porque ele está presente em indústrias de diferentes áreas, como têxteis, químicas, de tintas, de papéis e de semicondutores (Franca et al., 1996), (Rabadi et al., 2006), (Randhawa and Kuo, 1997). Além disso, o problema abordado também tem importância teórica, visto que ele pertence à classe de problemas $\mathcal{NP}$-difíceis, uma vez que ele é uma generalização do problema de sequenciamento em máquinas idênticas e sem tempos de preparação. A comprovação sobre pertencer a esta classe de problemas foi realizada em Garey and Johnson (1979) e Karp (1972).

Neste trabalho é abordada uma versão do problema no contexto de *green scheduling*, recentemente proposta em Cota et al. (2018). Os objetivos neste problema são minimizar o *makespan* e o consumo total de energia. Naquele trabalho é comprovado o conflito existente entre os dois objetivos e a importância de se considerar o consumo total de energia no problema.

O problema tem as características descritas a seguir. Existem um conjunto não-preemptivo de tarefas, um conjunto de máquinas paralelas e independentes e um conjunto de modos de operação para cada máquina. Cada modo de operação está relacionado a uma velocidade de operação e a um consumo de energia. Os objetivos são alocar todas as tarefas nas máquinas usando os modos de operação, buscando minimizar o *makespan* e o consumo total de energia. Segundo a notação de Graham et al. (1979) para problemas de sequenciamento em máquinas, este problema pode descrito como $R_M|S_{ijk}|(C_{\max}, TEC)$. Nessa descrição, $R_M$ representa os tempos de processamento, $S_{ijk}$ os tempos de preparação, $C_{\max}$ o *makespan* e TEC o consumo total de energia.

Na literatura são encontrados diversos trabalhos abordando a versão mono-objetivo do problema que busca minimizar apenas o *makespan* (definida por $R_M|S_{ijk}|C_{\max}$). Em Rabadi et al. (2006), os autores propõem uma meta-heurística chamada RAPS e um modelo matemático. Dois algoritmos genéticos e um modelo matemático são propostos em Vallada and Ruiz (2011). Vários algoritmos que combinam as meta-heurísticas *Iterated Local Search* e *Variable Neighborhood Descent* (Hansen et al., 2008) foram propostos em Cota et al. (2014) e Haddad et al. (2014). Em Cota et al. (2017) é proposto um algoritmo baseado na meta-heurística *Adaptive Large Neighborhood Search* (ALNS) (Ropke and Pisinger, 2006) com aprendizado em autômatos.

Apesar da extensa aplicabilidade dos problemas de sequenciamento de tarefas em máquinas, o tema *green scheduling* ainda é pouco explorado nestes problemas. Este tema envolve o desenvolvimento de abordagens sustentáveis para problemas de sequenciamento (Bampis et al., 2015), (Mansouri et al., 2016a), (Mansouri et al., 2016b), (Yu, 2010). Os trabalhos de Mansouri et al. (2016a) e de Mansouri and Aktas (2016) são exemplos disso, pois em ambos são tratados problemas de *flow shop* com os objetivos de minimizar o *makespan* e o consumo total de energia.

Em Cota et al. (2018) o problema biobjetivo $R_M|S_{ijk}|(C_{\max}, TEC)$ é resolvido apenas com o uso de métodos exatos. Na prática, este problema pode aparecer com diferentes tamanhos e o uso de métodos exatos pode ser inviável em problemas de grande porte em um tempo restrito. O objetivo principal deste trabalho é propor uma meta-heurística adaptativa capaz de resolver eficientemente o problema tratado, e que tenha aplicabilidade prática. Inspirado no algoritmo ALNS com aprendizagem em autômatos proposto em Cota et al. (2017), que resolve de maneira eficiente a versão mono-objetivo do problema, é proposto um algoritmo ALNS multiobjetivo para resolver o problema $R_M|S_{ijk}|(C_{\max}, TEC)$.

O restante deste trabalho está organizado como segue. A Seção 2 apresenta a definição do problema. O algoritmo proposto é descrito na Seção 3. Na Seção 4 os experimentos computacionais são reportados e discutidos. Finalmente, na Seção 5, as conclusões e os trabalhos futuros são apresentados.

## 2 Definição do problema

O modelo matemático para o problema $R_M|S_{ijk}|(C_{\max}, \mathit{TEC})$ é aquele de Cota et al. (2018). Para defini-lo, sejam:

- $M = \{1, ..., m\}$: o conjunto de máquinas, sendo $m$ o número de máquinas;
- $N = \{1, ..., n\}$: o conjunto de tarefas, sendo $n$ o número de tarefas;
- $L = \{1, ..., o\}$: o conjunto de $o$ diferentes modos de operação, sendo $o$ o número de modos de operação. Cada modo é relacionado a uma velocidade correspondente de operação e um consumo de energia elétrica;
- $p_{ik}$: o tempo de processamento da tarefa $k$ na máquina $i$ [em minutos];
- $S_{ijk}$: o tempo de preparação necessário para o processamento da tarefa $k$ na máquina $i$ após a tarefa $j$ [em minutos];
- $\pi_i$: potência da máquina $i$ em velocidade normal de operação [kW];
- $v_l$: fator de multiplicação de velocidade em operação normal, com $l \in L$;
- $\lambda_l$: fator de multiplicação de energia em velocidade normal, com $l \in L$;
- $B$: uma constante com valor elevado.

O fator $v_l$ é uma função não decrescente de $\lambda_l$, e é aplicado a todas as máquinas. A relação entre eles é apresentada como segue.

- Se $v_l = 1$ e $\lambda_l = 1$ indica que a máquina está na velocidade normal de operação;
- Se $0 < v_l < 1$ e $0 < \lambda_l < 1$ indica velocidade mais devagar que a normal, então a máquina consome menos energia;
- Se $v_l > 1$ e $\lambda_l > 1$ indica velocidade maior que a normal, então a máquina consome mais energia.

As variáveis de decisão usadas no modelo matemático são:

- $x_{ijkl}$: 1, se a tarefa $k$, com o modo de operação $l$, é alocada imediatamente após a tarefa $j$ na máquina $i$. 0, caso contrário;
- $C_j$: tempo de conclusão da tarefa $j$;
- $O_i$: tempo de conclusão da máquina $i$;
- $C_{\max}$: tempo máximo de processamento de todas as máquinas (*makespan*);
- *TEC*: consumo total de energia [kWh].

Uma tarefa fictícia 0 é utilizada no modelo, sendo ela alocada no início de cada máquina e possui tempo de processamento e preparação iguais a 0 ($p_{i0} = 0$ $\forall i \in M$ e $S_{i0k} = 0$ $\forall i \in M, \forall k \in N$). O modelo matemático é descrito pelas Equações (1) a (12):

$$\min C_{\max} \tag{1}$$

$$\min \textit{TEC} \tag{2}$$

Sujeito a:

$$\sum_{i=1}^{m}\sum_{\substack{j=0\\j\neq k}}^{n}\sum_{l=1}^{o} x_{ijkl} = 1 \quad \forall k \in N \tag{3}$$

$$\sum_{i=1}^{m}\sum_{\substack{k=1\\j\neq k}}^{n}\sum_{l=1}^{o} x_{ijkl} \leq 1 \quad \forall j \in N \tag{4}$$

$$\sum_{k=1}^{n}\sum_{l=1}^{o} x_{i0kl} \leq 1 \quad \forall\, i \in M \tag{5}$$

$$\sum_{\substack{k=0\\k\neq j}}^{n}\sum_{l=1}^{o} x_{ijkl} - \sum_{\substack{h=0\\h\neq j}}^{n}\sum_{l=1}^{o} x_{ihjl} = 0$$
$$\forall j \in N, \forall i \in M \tag{6}$$

$$C_k - C_j + B\left(1 - x_{ijkl}\right) \geq S_{ijk} + \frac{p_{ik}}{v_l}$$
$$\forall j \in N_0, \forall k \in N, j \neq k, \forall l \in L, \forall i \in M \tag{7}$$

$$C_0 = 0 \tag{8}$$

$$\sum_{j=0}^{m}\sum_{\substack{k=1\\k\neq j}}^{n}\sum_{l=1}^{o}\left(S_{ijk} + \frac{p_{ik}}{v_l}\right) x_{ijkl} = O_i$$
$$\forall i \in M \tag{9}$$

$$C_{\max} \geq O_i \quad \forall i \in M \tag{10}$$

$$\textit{TEC} \geq \sum_{i=1}^{m}\sum_{j=0}^{n}\sum_{\substack{k=1\\j\neq k}}^{n}\sum_{l=1}^{o}\left(\lambda_l \times \frac{\pi_i}{60} \times \frac{p_{ik}}{v_l}\right) x_{ijkl} \tag{11}$$

$$x_{ijkl} \in \{0,\, 1\} \quad \forall j \in N_0, \forall k \in N, j \neq k,$$
$$\forall i \in M, \forall l \in L \tag{12}$$

As funções objetivo correspondem à minimização do *makespan* (1) e do consumo total de energia (2). As restrições (3) definem que cada tarefa será alocada em apenas uma máquina e terá uma antecessora. Essas restrições também definem que há alocação de um único modo de operação. As restrições (4) e (5) asseguram que cada tarefa terá, no máximo, uma tarefa sucessora. As restrições (6) definem a ordem correta para a alocação das tarefas. As restrições (7) calculam o tempo acumulado para cada tarefa. Se $x_{ijkl} = 1$, o tempo acumulado de $k$ é $C_j$ mais $S_{ijk}$ e $\left(\frac{p_{ik}}{v_l}\right)$. Se $x_{ijkl} = 0$, a constante $B$ irá garantir que as restrições sejam satisfeitas. A restrição (8) define que o tempo acumulado da tarefa fictícia é igual a 0. As restrições (9) definem o cálculo dos custos acumulados para cada máquina, representado pela soma do tempo de preparação e do tempo de processamento de todas as tarefas alocadas numa dada máquina. As restrições (10) definem o valor do *makespan* (ou $C_{\max}$). A restrição (11) define o cálculo do total de energia consumido (ou *TEC*). Esse cálculo utiliza os tempos de processamento ($p_{ik}/v_l$), o consumo de potência de cada máquina na velocidade normal de operação ($\pi_i$) e os fatores de multiplicação ($\lambda_l$) e ($v_l$). Finalmente, as restrições (12) definem quais variáveis são binárias. Esse modelo matemático possui $n^2mq$ variáveis binárias, $n + m + 2$ variáveis contínuas e $2n + 3m + nm + 2n^2mq + 2$ restrições.

## 3 Algoritmo proposto

Esta Seção apresenta o algoritmo multiobjetivo proposto. Inicialmente, a representação e a avaliação da solução são apresentadas. Em seguida, o algoritmo multiobjetivo é detalhado.

### *3.1 Representação e avaliação da solução*

Uma solução é representada por duas estruturas de dados. Uma estrutura é utilizada para representar a alocação das tarefas nas máquinas e a outra é utilizada para representar o modo de operação no qual cada tarefa é processada. Cada modo de operação é relacionado a uma velocidade de operação e a um consumo de energia.

Um vetor de inteiros com $m$ posições é utilizado para representar a alocação das tarefas, sendo que $m$ é o número total de máquinas. Uma lista é associada a cada posição do vetor e isso representa as tarefas alocadas em cada máquina. Os modos de operação são representados por um vetor de inteiros. A Figura 1 mostra uma possível solução para uma instância com duas máquinas, sete tarefas e três modos de operação. As tarefas 7, 3, e 4 estão alocadas na máquina $M_1$, nesta ordem. As tarefas 2, 1, 6 e 5 estão alocadas na máquina $M_2$, nesta ordem. Os modos de operação são definidos como segue : $i$) modo de operação 1: tarefas 3 e 4; $ii$) modo de operação 2: tarefas 1, 5 e 7; $iii$) e modo de operação 3: tarefas 2 e 6.

A solução é avaliada a partir do *makespan* (como definido na Equação (10), Seção 2) e o total de consumo de energia (como definido na Equação (11), Seção 2).

### *3.2 ALNS multiobjetivo*

O algoritmo proposto é uma versão multiobjetivo do ALNS com aprendizado em autômatos (ou *Learning Automata* – LA) definido em Cota et al. (2017). Além disso, ambos os algoritmos possuem características em comum. É definida uma *Learning Automata* para os métodos de remoção ($LA_{N^-}$) e outra para os métodos de inserção ($LA_{N^+}$). Uma heurística baseada

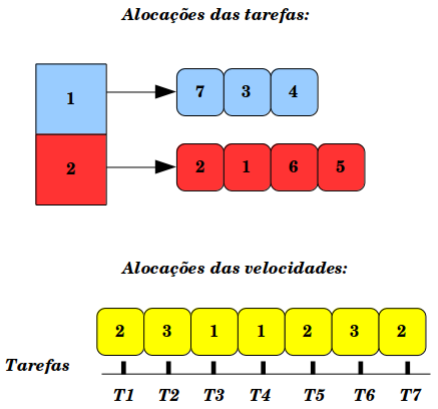


Figura 1: Representação de uma solução.

na fase construtiva da meta-heurística GRASP (Feo and Resende, 1995) é utilizada para a geração da solução inicial. Um método multiobjetivo de descida em vizinhança variável aleatória (MO-RVND) é utilizado para realizar as buscas locais.

As equações do LA que definem as atualizações conforme a resposta do processo são apresentadas a seguir. Caso a reposta seja satisfatória, as probabilidades das ações são atualizadas de acordo com a Equação (13). Caso contrário, a Equação (14) é utilizada para atualizar as probabilidades (Vafashoar and Meybodi, 2016).

$$p_y(w+1) = \begin{cases} p_y(w) + a(1 - p_y(w)) & \text{se } z = y \\ p_y(w)(1-a) & \text{se } z \neq y \end{cases} \quad (13)$$

$$p_y(w+1) = \begin{cases} p_y(w)(1-b) & \text{se } z = y \\ \frac{b}{r-1} + p_y(w)(1-b) & \text{se } z \neq y \end{cases} \quad (14)$$

Para as Equações (13) e (14), $a$ e $b$ são definidos como parâmetros de *recompensa* e *penalidade*, respectivamente. O índice $z$ indica o método da LA que foi aplicado no momento, o índice $y$ representa todos os métodos da LA e o índice $w$ representa a iteração corrente.

O pseudocódigo do ALNS multiobjetivo é apresentado no Algoritmo 1.

No Algoritmo 1, os parâmetros $a_1$, $a_2$ e $a_3$ são recompensas e o parâmetro $b_1$ é a penalidade. O parâmetro $q$ define o número de tarefas que são removidas e inseridas na solução corrente a cada iteração do algoritmo. O parâmetro $K$ define a periodicidade em que as probabilidade são atualizadas. O tempo máximo de execução do algoritmo é definido por $t_{max}$ e ele é utilizado como critério de parada. Os valores dos parâmetros foram calibrados em Cota et al. (2017), e os mesmos valores são utilizados neste trabalho. Os valores são: *i)* $k$: $6 \times \max(|\alpha^-|, |\alpha^+|)$; *ii)* $q$: 5% do número de tarefa; *iii)* $a_1 = 0.2$; *iv)* $a_2 = 0.1$; *v)* $a_3 = 0.05$; e *vi)* $b_1 = 0.02$. Os valores de recompensa e penalidade são baseados nas discussões apresentadas em Vafashoar and Meybodi (2016).

**Algoritmo 1:** ALNS multiobjetivo

**entrada:** $K, q, a_1, a_2, a_3, b_1, t_{max}$
**saída** : $s_{best}$

Defina LA $(\phi^-, \alpha^-, \beta, A^-, \pi^-, p^-)$;
Defina LA $(\phi^+, \alpha^+, \beta, A^+, \pi^+, p^+)$;
$s \leftarrow$ procedimentoConstrutivo ();
$D \leftarrow$ adicionaSolução (*D,s,f*) ; /* Conjunto de soluções não-dominadas */
$T \leftarrow (0,05 \times (\sum_{r=1}^{R} 0,5 \times f_r(s)))/\ln 2$;
**enquanto** $tempoCorrente \leq t_{\max}$ **faça**
  Selecione um método de remoção $\alpha_i^-$ e um método de inserção $\alpha_j^+$ utilizando o método da roleta;
  Remova $q$ tarefas de $s$ utilizando $\alpha_i^-$;
  Insira as tarefas removidas em $s$ utilizando $\alpha_j^+$;
  $s' \leftarrow$ buscaLocal $(s)$;
  **se** $(f(s') \prec f(s))$ **então**
    $D \leftarrow$ adicionaSolução (*D,s',f*);
    $a \leftarrow a_1; b \leftarrow 0$;
    Atualize $p^-$ e $p^+$ usando as Eq. (13)-(14);
  **fim**
  **senão se** $(f(s') \not\prec f(s)) \wedge (f(s) \not\prec f(s'))$ **então**
    $D \leftarrow$ adicionaSolução (*D,s',f*);
    $a \leftarrow a_2; b \leftarrow 0$;
    Atualize $p^-$ e $p^+$ usando as Eq. (13)-(14);
  **fim**
  **senão se** aleatório $< \min\left\{1, \exp\left(\frac{\sum_{r=1}^{R} 0,5 \times f_r(s')}{T}\right)\right\}$ **então**
    $s'' \leftarrow s'$;
    $a \leftarrow a_3; b \leftarrow 0$;
    Atualize $p^-$ e $p^+$ usando Eq. (13)-(14);
  **fim**
  **senão**
    $a \leftarrow 0; b \leftarrow b_1$;
    Atualize $p^-$ e $p^+$ usando as Eq. (13)-(14);
  **fim**
  Seleção aleatória de $s$ em $(D, s'')$;
  **se** $k \mod K = 0$ **então**
    atualizaLA$(A^-, A^+, p^-, p^+)$;
  **fim**
  $T \leftarrow 0,99 \times T$;
**fim**
**retorna** $D$

Inicialmente, uma solução $s$ é construída, e um conjunto de soluções não-dominadas $D$ é criado. O conjunto $D$ é atualizado pelo método `adicionaSolução`. Esse método utiliza o critério de dominância Pareto para avaliar as soluções. Para mais informações sobre esse procedimento, sugere-se a leitura de Lust and Teghem (2010). A temperatura inicial é calculada de modo que a solução corrente tenha 50% de chance de aceitação, se ela é 5% pior que a solução inicial. É considerada a soma ponderada dos objetivos da solução inicial, sendo que cada objetivo tem peso igual a 0,5.

Os passos do ALNS multiobjetivo a cada iteração são:

1. Um método de remoção $\alpha_i^- \in \alpha^-$ e um método de inserção $\alpha_j^+ \in \alpha^+$ são selecionados por um método do tipo roleta, usando as probabilidades de cada um deles;

2. $q$ tarefas são removidas da solução corrente com o método $\alpha_i^-$ e reinseridas com o método $\alpha_j^+$;

3. A busca local RVND é aplicada à solução $s'$;

4. As probabilidade $p^-$ e $p^+$ são atualizadas. Se a solução $s'$ for aceita, a Equação (13) é utilizada. Caso contrário, se a solução $s'$ não for aceita, a Equação (14) é aplicada. Se a solução corrente $s'$ domina a solução $s$ ou não existe dominância entre elas, uma tentativa de inserir $s'$ no conjunto

$D$ usando o método `adicionaSolução` é realizada. Se a solução corrente é aceita conforme o critério de temperatura, a solução $s''$ recebe a solução $s'$;

5. Uma nova solução $s$ é selecionada aleatoriamente dentre todas as soluções do conjunto $D$ e a solução $s''$;

6. A cada $K$ iterações, os valores das probabilidades de seleção são atualizados em cada LA;

7. Ao final, o conjunto $D$ de soluções não-dominadas é retornado.

Uma solução pode ser aceita em três diferentes situações e, para cada uma delas, um parâmetro de recompensa ($a_1$, $a_2$ ou $a_3$) é utilizado na atualização dos LA. As situações são: $i$) $a_1$: se a solução $s'$ domina $s$; $ii$) $a_2$: se não existe dominância entre as soluções $s'$ e $s$; $iii$) $a_3$: se a solução $s'$ é dominada por $s$, mas é aceita segundo o critério de temperatura. Se a solução encontrada não é aceita, o parâmetro de penalidade ($b_1$) é utilizado para atualizar os LA.

### 3.2.1 Procedimento construtivo

Este processo é inspirado na fase construtiva da heurística GRASP e funciona como segue. A heurística tem um limite de tempo igual a 1% do valor de $t_{\max}$. Inicialmente, uma operação $l \in L$ é selecionada aleatoriamente (conforme representado no vetor de inteiros da Figura 1) para cada alocação. Uma nova solução é gerada a cada iteração da heurística e apenas a melhor solução é armazenada. Novas soluções são geradas por meio de uma heurística semigulosa baseada no método *Adaptive Shortest Processing Time* (ASPT) (Baker, 1974).

Na construção semigulosa, todas as tarefas são inseridas em uma lista de candidatos $LC$. A cada iteração é avaliada a inserção de cada tarefa $j \in LC$ no final de cada máquina $i \in M$, considerando os tempos de processamento e preparação. Uma segunda lista $best_{LC}$ armazena 20% das alocações que geram um menor custo à máquina. Após isso, uma tarefa $k$ é selecionada aleatoriamente em $best_{LC}$ e é inserida na máquina correspondente. A tarefa $k$ também é removida de $LC$. Esse processo é repetido até todas as tarefas de $LC$ sejam alocadas.

### 3.2.2 Heurísticas de inserção e remoção

Os métodos de inserção e remoção utilizados no ALNS multiobjetivo são os mesmos propostos em Cota et al. (2017) para o ALNS com *Learning Automata*. Os métodos de remoção são apresentados na Tabela 1. Todos os métodos removem $q$ tarefas.

Os métodos de inserção são apresentados na Tabela 2. As $q$ tarefas removidas são embaralhadas antes da execução em um dos métodos de inserção.

Tabela 1: Métodos de remoção

| Método | Ação |
|---|---|
| Remoção aleatória | Remove tarefas aleatoriamente. |
| Remoção gulosa do maior custo | Remove tarefas com maiores custos, considerando os tempos de preparação e processamento. |
| Remoção semigulosa do maior custo | Esta é uma versão semigulosa do método anterior. A cada iteração, uma tarefa dentre as 20% com maior custo é removida aleatoriamente. |
| Remoção de máquina aleatória | Remove tarefas de uma máquina selecionada aleatoriamente. |
| Remoção da máquina com maior custo | Remove tarefas da máquina com o maior custo. |
| Remoção *Shaw* | Remove tarefas que são consideradas similares. O critério de similaridade é dado pela soma do custo de processamento e o tempo de preparação de todas as tarefas alocadas. |

Tabela 2: Métodos de inserção

| Método | Ação |
|---|---|
| Inserção gulosa | Insere a tarefa na melhor posição, considerando todas as posições de todas as máquinas. |
| Inserção semigulosa | Esta é uma versão semigulosa do método anterior. A cada iteração, uma tarefa é inserida numa posição aleatória dentre as 20% das melhores posições. |
| Inserção *Lambda* | Insere 50% das tarefas utilizando o método semiguloso e os outros 50% são inseridos aleatoriamente. |
| Inserção ILS | Para cada tarefa, uma máquina é selecionada aleatoriamente e a tarefa é inserida na melhor posição dessa máquina. |
| Inserção por arrependimento | Insere primeiro as tarefas com maior custo de arrependimento. Esse custo é dado pela diferença entre a melhor posição e a segunda melhor posição, considerando todas as máquinas. |
| Inserção pelo Método Húngaro | Uma matriz quadrada é criada, sendo que as linhas são as máquinas $m \in M$ e as colunas são as $m$ primeiras tarefas de $q$. A matriz é preenchida com o melhor custo de alocação para cada tarefa na máquina. O Método Húngaro é utilizado para encontrar a melhor alocação. Esse processo é repetido até que todas as tarefas de $q$ sejam inseridas. |

### 3.2.3 Procedimentos de busca local

Uma versão multiobjetivo do algoritmo de descida em vizinhança variável aleatória (ou *random variable neighborhood descent multiobjective*) (Souza et al., 2010) é empregada como procedimento de busca local. A contrário de sua versão clássica *variable neighborhood descent* (Hansen et al., 2008), este método tem a vantagem de não necessitar da calibragem da melhor ordem das vizinhanças, já que a melhor ordem para aplicar as buscas locais pode não depender apenas do problema abordado, mas também das características das instâncias. Foram definidas quatro estruturas de vizinhança para serem utilizadas pelo procedimento, como descrito a seguir.

1. **Inserção múltipla –** $NS^{MI}(s)$: movimento que realoca uma tarefa de uma máquina em outra posição na mesma máquina, ou que realoca uma tarefa numa posição qualquer em outra máquina.

2. **Troca na mesma máquina –** $NS^{SSM}(s)$: movimento de troca (*swap*) de duas tarefas na mesma máquina.

3. **Troca entre máquinas diferentes –** $NS^{SDM}(s)$: movimento de troca de duas tarefas em máquinas diferentes.

4. **Troca de modos de operação –** $NS^{SMO}(s)$: movimento de troca do modo de operação de uma alocação.

O espaço de busca é explorado por meio de quatro procedimentos de busca local, os quais utilizam as estruturas de vizinhança descritas anteriormente. Todas as buscas locais usam como critério de parada a estratégia de primeira melhora (*first improvement strategy*).

Três das buscas locais foram utilizadas em trabalhos anteriores, sendo: $i)FI_{MI}$ (Cota et al., 2014; Cota et al., 2017): utiliza a estrutura de vizinhança $NS^{MI}$; $ii)$ $FI_{SDM}$ (Haddad et al., 2014; Cota et al., 2017): a estrutura de vizinhança $NS^{SDM}$ é utilizada; e $iii)$ $FI_{SSM}$ (Cota et al., 2017): é empregada a estrutura de vizinha $NS^{SSM}$. O critério de aceitação utilizado neste trabalho emprega as relações de dominância Pareto. A solução corrente $s'$ é aceita caso ela domine a solução $s$ ou se não existe dominância entre elas. Essa regra também é aplicada como critério de aceitação para o MO-RVND.

A quarta busca local é chamada de $FI_{SMO}$ e ela utiliza a estrutura de vizinhança $NS^{SMO}$. O procedimento é realizado como segue. Inicialmente, todas as máquinas são classificadas em ordem decrescente pelo tempo de conclusão. As máquinas são selecionadas do maior para o menor tempo de conclusão. Para cada máquina, todas as trocas dos modos de operação entre as suas tarefas são avaliadas. Se um vizinho $s' \in NS^{SSM}$ domina a solução $s$ ou não existe dominância entre elas, a solução $s'$ é aceita e a pesquisa termina. Caso contrário, o processo continua até que todas as máquinas sejam analisadas.

A seguir, os experimentos computacionais realizados para validar o algoritmo proposto.

## 4 Experimentos computacionais

O algoritmo ALNS multiobjetivo foi implementado na linguagem JAVA com a IDE NetBeans 8.0.2. Os experimentos computacionais foram realizados em um computador Intel Core i7 com 1,9 GHz, 6GB de RAM e sistema operacional Ubuntu 16.04.

### *4.1 Conjunto de instâncias*

Para realizar os experimentos computacionais foi utilizado um conjunto de instâncias proposto em Cota et al. (2018). Este conjunto contém 80 instâncias [1] com as seguintes características: $i)$ número de tarefas: 6, 8, 10, 12 e 15; $ii)$ número de máquinas: 2, 3, 4 e 5; $iii)$ número de modos de operação: 3 ou 5; $iv)$ tempos de processamento: $U[1, 99]$; $v)$ tempo de preparação: $U[1, 9]$, $U[1, 49]$, $U[1, 99]$ e $U[1, 124]$ e $vi)$ potência da máquina (Kw): $U[40, 200]$. Este conjunto de instâncias é disponibilizado em `http://www.decom.ufop.br/prof/marcone/projects/upmsp/Small%20Instances%20%20-%20RM%20%7CSijk%7C(Cmax%20,%20TEC).zip`.

### *4.2 Comparação entre os resultados do ALNS multiobjetivo e do $\epsilon$-restrito*

Os resultados do algoritmo ALNS multiobjetivo são comparados aos do método $\epsilon$-restrito (Ehrgott, 2005) disponíveis em Cota et al. (2018). O método $\epsilon$-restrito transforma um problema multiobjetivo em um monoobjetivo, criando restrições adicionais ao problema. Neste método um objetivo é selecionado e mantido na função objetivo do problema, os outros objetivos são transformados em restrição.

O tempo limite (ou $t_{\max}$) é definido pela Eq. 15, sendo que $n$ é o número de tarefas. Este critério foi inspirado nos utilizados em Cota et al. (2017) e Cota et al. (2014) para tratar um problema semelhante ao abordado neste trabalho.

$$t_{\max} = n \times 50 \ \ segundos \tag{15}$$

A métrica utilizada para comparar os resultados do ALNS multiobjetivo e do método $\epsilon$-restrito é o hipervolume. O hipervolume de um conjunto $P$ é dado pelo volume da região entre os pontos $po_i \in P$ e um ponto de referência. Para cada solução é construído um hipercubo de acordo com o ponto de referência. Em problemas de minimização, como é o caso deste trabalho, o ponto de referência é um limite superior para cada objetivo. Já para problemas de maximização o ponto de referência geralmente utilizado é $(0, 0)$. O hipervolume de uma estimativa de fronteira Pareto é a soma dos hipercubos que cada conjunto de soluções contém. Neste trabalho os pontos de referências utilizados são os mesmos usados em Cota et al. (2018). Os valores do hipervolume foram normalizados entre 0 e 1, dividindo o somatório dos hipercubos pela área máxima que contém a Pareto.

Como o algoritmo ALNS multiobjetivo tem natureza estocástica, ele foi executado cinco vezes para cada instância. Assim, apenas o valor médio do hipervolume é considerado nos resultados.

A Tabela 3 apresenta os resultados médios dos métodos e a informação se existe diferença estatística. Os resultados estão agrupados pelas instâncias com o mesmo número de tarefas. Por exemplo, a primeira linha indica os resultados médios para todas as instâncias que possuem 6 tarefas (total de 16 instâncias), ou seja, incluindo todas as combinações de máquinas e modos de operação. Já a Tabela 4 apresenta os valores máximos, mínimos e mediana destes resultados.

Tabela 3: Comparação dos resultados médios e do teste estatístico para o algoritmo ALNS multiobjetivo e do método $\epsilon$-restrito.

| Conjunto de instâncias | $\epsilon$-restrito | ALNS multiobjetivo | Há diferença estatística? |
|---|---|---|---|
| 6 | 0,708 ($\pm$ 0,086) | **0,817** ($\pm$ 0,021) | Sim |
| 8 | 0,768 ($\pm$ 0,093) | **0,822** ($\pm$ 0,113) | Sim |
| 10 | 0,791 ($\pm$ 0,055) | **0,825** ($\pm$ 0,008) | Sim |
| 12 | **0,863** ($\pm$ 0,093) | 0,841 ($\pm$ 0,129) | Sim |
| 15 | **0,833** ($\pm$ 0,088) | 0,799 ($\pm$ 0,052) | Não |

Os melhores resultados médios da Tabela 3 estão destacados em negrito e para cada resultado é apresentado o desvio padrão entre parênteses. Esses resultados sugerem que o algoritmo ALNS multiobjetivo encontrou os melhores resultados na maioria dos casos para o ambiente experimental utilizado. Os resultados

[1] As características completas deste conjunto de instâncias podem ser encontradas em Cota et al. (2018).

Tabela 4: Comparação dos valores máximos, mínimo e mediana do algoritmo ALNS multiobjetivo e do método $\epsilon$-restrito.

| Conjunto de instâncias | $\epsilon$-restrito | | | ALNS multiobjetivo | | |
|---|---|---|---|---|---|---|
| | Max | Min | Mediana | Max | Min | Mediana |
| 6 | 0,861 | 0,536 | 0,706 | 0,950 | 0,706 | 0,810 |
| 8 | 0,881 | 0,511 | 0,774 | 0,923 | 0,568 | 0,843 |
| 10 | 0,883 | 0,697 | 0,792 | 0,907 | 0,726 | 0,840 |
| 12 | 0,939 | 0,619 | 0,838 | 0,948 | 0,855 | 0,688 |
| 15 | 0,931 | 0,615 | 0,788 | 0,878 | 0,814 | 0,666 |

do algoritmo também sugerem um boa convergência para a fronteira Pareto, já que os resultados para a métrica hipervolume são similares aos obtidos pelo método exato.

O teste estatístico aplicado na Tabela 3 é o teste não-paramétrico sinal (Sheskin, 2006) com significância de $0,05$. Para cada grupo de instâncias (com o mesmo número de tarefas) existem 80 amostras para o algoritmo estocástico ALNS (5 execuções para cada uma das 16 instâncias) e 16 amostras para o método determinístico $\epsilon$-restrito (uma execução para cada instância). O teste sinal indica que o ganho de um método deve ser em no mínimo $\frac{n}{2} + 1,96 \times \frac{\sqrt{n}}{2}$ casos, como temos $n = 80$ em cada grupo, o resultado da equação é 48,76. Analisando os resultados completos, observa-se que o ALNS foi o melhor em mais de 48,76 casos em quatro grupos de instâncias.

Uma análise gráfica da convergência do algoritmo ALNS multiobjetivo também foi realizada. Para isso, foi selecionada aleatoriamente uma instância para executar a avaliação. A Figura 2 mostra os resultados.

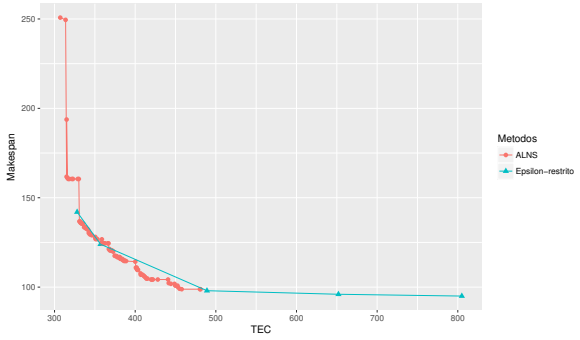


Figura 2: Fronteira Pareto obtida para uma instância com 12 tarefas e 4 máquinas. Os eixos representam os dois objetivos do problema.

Os resultados apresentados pela figura também evidenciam que o algoritmo ALNS multiobjetivo tem uma boa convergência para a fronteira Pareto, considerando o ambiente experimental utilizado. Pode-se observar também que o ALNS é capaz de encontrar soluções com até 550 unidades de consumo de energia [kwh], evidenciando a importância do problema e da abordagem proposta.

## 5 Conclusões

Este trabalho abordou o problema de sequenciamento em máquinas paralelas não relacionadas com tempos de preparação dependentes da sequência. Os objetivos são a minimização do *makespan* e o consumo total de energia.

Esta versão do problema foi recentemente proposta na literatura e até então só foram analisados métodos exatos para a sua resolução. Os métodos exatos têm limitação na resolução de problemas de grande porte em um tempo restrito. Por isso, o objetivo principal deste trabalho foi propor um algoritmo heurístico adaptativo para resolver o problema. O algoritmo proposto é um ALNS multiobjetivo que utiliza aprendizagem em autômatos para aperfeiçoar o processo de busca.

Nos experimentos computacionais foram utilizados conjuntos de instâncias de pequeno e médio porte da literatura. O algoritmo ALNS multiobjetivo foi executado para todas as instâncias e os seus resultados foram comparados aos do método exato $\epsilon$-restrito. Na comparação foram utilizados o indicador hipervolume e análise gráfica. Os resultados sugerem que o algoritmo proposto possui uma boa convergência para a fronteira Pareto. Devido a natureza adaptativa do algoritmo proposto, ele pode ser aplicado em instâncias reais do problema em questão.

Como trabalhos futuros, propõem-se a criação de um conjunto de instâncias de grande porte para o problema e a resolução destas instâncias usando o algoritmo ALNS multiobjetivo e outros algoritmos de otimização multiobjetivo.

## Agradecimentos

Os autores agradecem às agências FAPEMIG, CNPq e CAPES, bem como à UFOP, à UFMG e ao ITV pelo apoio ao desenvolvimento deste trabalho.

## Referências

Baker, K. R. (1974). *Introduction to Sequencing and Scheduling*, John Wiley & Sons, Inc.

Bampis, E., Letsios, D. and Lucarelli, G. (2015). Green scheduling, flows and matchings, *Theoretical Computer Science* **579**: 126 – 136.

Cota, L. P., Coelho, V. N., Guimarães, F. G. and Souza, M. J. F. (2018). Bi-criteria formulation for green scheduling with unrelated parallel machines with sequence-dependet setup times, *International Transactions in Operation Research* .

Cota, L. P., Guimarães, F. G., Oliveira, F. B. and Souza, M. J. F. (2017). An adaptive large neighborhood search with learning automata for the unrelated parallel machine scheduling problem,

*Proceedings of the 2017 IEEE Congress on Evolutionary Computation (CEC 2017)*, Donóstia - San Sebastian, pp. 185–192.

Cota, L. P., Haddad, M. N., Souza, M. J. F. and Coelho, V. N. (2014). AIRP: A heuristic algorithm for solving the unrelated parallel machine sheduling problem, *Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC 2014)*, Beijing, pp. 1855–1862.

Ehrgott, M. (2005). *Multicriteria Optimization*, 3nd edn, Springer Publishing Company, Incorporated.

Feo, T. and Resende, M. (1995). Greedy randomized search procedures, *Journal of Global Optimization* **6**(2): 109–133.

Franca, P. M., Gendreau, M., Laporte, G. and Muller, F. M. (1996). A tabu search heuristic for the multiprocessor scheduling problem with sequence dependent setup times, *International Journal of Production Economics* **43**: 79–89.

Garey, M. and Johnson, D. (1979). Computers and intractability: A guide to the theory of NP-completeness, *WH Freeman & Co., San Francisco* **174**.

Graham, R., Lawler, E., Lenstra, J. and Kan, A. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey, *Annals of discrete Mathematics* **5**(2): 287–326.

Haddad, M. N., Cota, L. P., Souza, M. J. F. and Maculan, N. (2014). AIV: A heuristic algorithm based on iterated local search and variable neighborhood descent for solving the unrelated parallel machine scheduling problem with setup times, *Proceedings of the 16th International Conference on Enterprise Information Systems (ICEIS 2014)*, Lisbon, Portugal, pp. 376–383.

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

Karp, R. M. (1972). Reducibility among combinatorial problems, *Complexity of Computer Computations* **40**(4): 85–103.

Lust, T. and Teghem, J. (2010). Two-phase pareto local search for the biobjective traveling, *Journal of Heuristics* **16**: 475–510.

Mansouri, S. A. and Aktas, E. (2016). Minimizing energy consumption and makespan in a two-machine flowshop scheduling problem, *Journal of the Operational Research Society* **67**(11): 1382–1394.

Mansouri, S. A., Aktas, E. and Besikci, U. (2016a). Green scheduling of a two-machine flow shop: Trade-off between makespan and energy consumption, *European Journal of Operation Research* **248**: 772–788.

Mansouri, S. A., Aktas, E. and Besikci, U. (2016b). Minimizing energy consumption and makespan in a two-machine flowshop scheduling problem, *Journal of the Operational Research Society* .

Rabadi, G., Moraga, R. J. and Al-Salem, A. (2006). Heuristics for the unrelated parallel machine scheduling problem with setup times, *Journal of Intelligent Manufacturing* **17**(1): 85–97.

Randhawa, S. U. and Kuo, C. H. (1997). Evaluating scheduling heuristics for non-identical parallel processors, *International Journal of Production Research* **35**: 969–981.

Ropke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows, *Transportation science* **40**: 455–472.

Sheskin, D. J. (2006). *Handbook of Parametric and Nonparametric Statistical Procedures*, Vol. 63, 4th edn, Chapman & Hall/CRC.

Souza, M., Coelho, I., Ribas, S., Santos, H. and Merschmann, L. (2010). A hybrid heuristic algorithm for the open-pit-mining operational planning problem, *European Journal of Operational Research* **207**(2): 1041–1051.

Vafashoar, R. and Meybodi, M. R. (2016). Multi swarm bare bones particle swarm optimization with distribution adaption, *Applied Soft Computing* **47**: 534 – 552.

Vallada, E. and Ruiz, R. (2011). A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times, *European Journal of Operational Research* **211**(3): 612–622.

Yu, J. J. (2010). Green scheduling and its solution, *Manufacturing Engineering and Automation I*, Vol. 139 of *Advanced Materials Research*, Trans Tech Publications, pp. 1415–1418.