

Modelo matemático e abordagens para o Problema de Sequenciamento em máquinas paralelas não relacionadas com restrição de recursos renováveis.

Vinicius Martins Ton (PPGMNE/GTAO/UFPR) vinnimt@gmail.com
Andressa Mendes Cardoso (PPGEP /GTAO/UFPR) andressa.adm@gmail.com
José Eduardo Pécora Jr (PPGMNE/PPGEP/GTAO/UFPR/CIRRELT) pecora@ufpr.br

Resumo:

Este trabalho estuda o Problema de Sequenciamento não Especificado e Dinâmico em Máquinas Paralelas não relacionadas com restrição de Recurso Renovável (UPMR). Um modelo de programação linear inteira é apresentado e, para facilitar a convergência da solução, são utilizadas inequações válidas, *Lower Bounds* e um *Upper Bound* advindo de uma heurística construtiva para delimitação do horizonte de tempo. O modelo foi testado em instâncias disponíveis na literatura e os resultados demonstraram a efetividade do modelo.

Palavras chave: Sequenciamento da Produção, Máquinas Paralelas, Recurso Renovável, Programação Inteira, Heurística Construtiva.

Mathematical model and approaches to unrelated parallel machine scheduling problem with renewable resources.

Abstract

This paper studies the Unspecified and Dynamic Unrelated Parallel Machine Scheduling Problem with Renewable Resources (UPMR). An integer linear programming is presented and, to make faster the convergence of the solution, we use valid inequalities, Lower Bounds and, as Upper Bound, a constructive heuristic for delimitation of the time horizon. The model was tested in instances available in the literature and the results demonstrated the effectiveness of the approach proposed.

Key-words: Scheduling Problem, Parallel Machine, Renewable Resource, Integer Programming, Constructive Heuristic.

1. Introdução

Milhares de empresas buscam diariamente obter um diferencial competitivo frente aos seus concorrentes, procurando desenvolver práticas que as coloquem em posição de vantagem no mercado em que atuam. Pela percepção dos clientes, inúmeros são os critérios que podem diferenciar uma empresa das demais. Preço e qualidade normalmente são as principais características procuradas pelos consumidores.

Para que os preços consigam ser competitivos, é importante que as empresas otimizem o uso dos seus recursos produtivos, buscando a excelência na gestão de operações. Desta forma, os custos fixos são diluídos em uma produção mais volumosa e, consequentemente, os preços

podem ficar mais atrativos para os clientes.

Neste sentido há um interesse crescente em estudos que visem dar apoio na tomada de decisão sobre Problemas de Programação da Produção, tais como adoção de sistemas de produção, ajustes de capacidade, dimensionamento de lotes, e sequenciamento de produção.

Este trabalho apresenta uma contribuição para a resolução de um Problema de Sequenciamento não Especificado e Dinâmico em Máquinas Paralelas não Relacionadas, com restrição de Recursos Renováveis (UPMR – *Unrelated Parallel Machine scheduling problem with a renewable Resource constraint*), que é uma variante do Problema de Sequenciamento em Máquinas Paralelas não Relacionadas (UPM - *Unrelated Parallel Machine scheduling problem*).

O UPM consiste na ordenação de um conjunto de tarefas em um conjunto de máquinas dispostas em paralelo, onde as tarefas podem ser processadas por qualquer uma das máquinas e o tempo de processamento das tarefas pode variar de uma máquina para outra. Normalmente o objetivo deste problema é a minimização do *makespan*, ou seja, terminar a última tarefa do conjunto no menor instante de tempo possível.

A variante estudada neste trabalho (UPMR) considera a utilização de recursos renováveis no processamento das tarefas (e.g. operadores, ferramentas, energia) que estão disponíveis em quantidade limitada e que, após sua utilização no processamento de uma tarefa, podem ser reutilizados. Isto torna o modelo mais próximo à realidade complexa das empresas de manufatura e possibilita uma solução mais adequada ao problema.

Para tanto, este artigo está estruturado da seguinte forma: na Seção 2 apresenta-se uma revisão de literatura afim de identificar o conhecimento existente até o momento sobre o Problema UPMR; na Seção 3 o problema é descrito e o MILP é apresentado; na Seção 4 são apresentados os *Lower* e *Upper Bounds* utilizados para facilitar a convergência do modelo; a Seção 5 traz os resultados computacionais; e, por fim, na Seção 6 são feitas algumas considerações sobre o trabalho.

2. Revisão de Literatura

Ventura e Kim (2000), em seu estudo sobre sequenciamento em máquinas paralelas idênticas, adicionam ao modelo original uma *due date*, ou seja, o instante de tempo em que o processamento de todas as tarefas deve terminar; e a possibilidade de utilização um tipo de recurso renovável (limitado a uma unidade por tarefa). O objetivo do modelo consiste na minimização da violação da *due date* para um determinado conjunto de tarefas. A partir de alguns Lemas e Teoremas, verificaram a possibilidade de simplificar o problema para uma adaptação do Problema de Designação assimétrico, eliminando a restrição que trata do recurso renovável e permitindo, assim, que os problemas sejam resolvidos em tempos computacionais aceitáveis.

Dando sequência ao estudo, Ventura e Kim (2003) passam a considerar um modelo mais complexo, onde mais de um tipo de recurso pode ser utilizado, mais de uma unidade de recurso pode ser requerida pelas tarefas e cada uma delas pode ter uma *due date* diferente das demais. Entretanto, neste modelo, todos os tempos de processamento das tarefas são iguais. O objetivo do modelo proposto é minimizar a soma dos desvios absolutos entre o tempo em que as tarefas terminam de ser processadas e suas respectivas *due dates*. Para tanto, é aplicada uma Relaxação Lagrangeana no modelo dual do problema com a finalidade de encontrar o maior *Lower Bound* para o problema primal. Também uma heurística construtiva é desenvolvida para encontrar uma solução inicial factível (*Upper Bound*).

Ruiz e Andrés (2007) apresentam uma outra variante do UPM, onde o tempo de *setup*, além de depender da máquina e da sequência das tarefas, depende também da quantidade de recurso adicional designado para seu processamento, de tal forma que quanto mais recurso alocado, menor o tempo de *setup*. O trabalho propõe um modelo de programação inteira para o RASDST (*Resource Assignable Sequence Dependent Setup Times*), cujo objetivo é minimizar o uso de recurso e o *makespan*. Ademais, propuseram três heurísticas construtivas para resolver o problema, cuja diferença entre elas consiste no critério de ordenação das tarefas na construção da solução. A heurística *Shortest Processing Time with Setups Resource Assignment* (SPTSA) ordena as tarefas pelo tempo de processamento; a heurística *Shortest Processing and Setup Time with Setups Resource Assignment* (SPSTSA) considera, além dos tempos de processamento, um valor médio de todos os possíveis valores para o tempo de *setup*; e a heurística *Dynamic Job Assignment with Setups Resource Assignment* (DJASA) constrói a solução de forma dinâmica, ordenando as tarefas em ordem crescente da função objetivo considerando o valor do tempo de *setup* daquela inserção.

Interessados no problema proposto por Ruiz e Andrés (2007), Kampke et al (2009) utilizam um GRASP Reativo com *Path Relinking* (RGPR) e Kampke et al (2010) utilizam um *Iterated Local Search* com *Path Relinking* (ILSPR), cuja solução inicial é construída pela DJASA (RUIZ E ANDRÉS, 2007). Ambos os trabalhos obtiveram resultados superiores em comparação com as heurísticas SPTSA, SPSTSA e DJASA para as instâncias maiores. Entretanto, não houve comparação entre RGPR e ILSPR.

Edis e Ozkarahan (2011a) pesquisam o problema de sequenciamento em máquinas paralelas idênticas com restrição de recurso e elegibilidade de máquinas (RCPMSP). Para tanto, utilizaram três abordagens. Um modelo de programação inteira (IP), um *constraint programming* (CP) e uma combinação destes dois. Foi também desenvolvida uma heurística para determinar um período factível eficiente (*initial feasible heuristic* – IFH), com o intuito de reduzir o tamanho do problema. Testes em 200 instâncias demonstraram que a combinação IP/CP com o algoritmo IFH obteve ótimos resultados

Edis e Ozkarahan (2011b) estudam uma aplicação real de um RCPMSP e propuseram duas abordagens de solução em que o problema é particionado em *loading* e *scheduling sub-problems*. Em ambas as abordagens, o problema de *loading* é resolvido pelo modelo exato e o que diferencia uma abordagem da outra é a utilização de programação inteira ou *constraint programming* na fase de *scheduling* – IP/IP ou IP/CP. Ambas as abordagens, assim como o modelo completo em IP foram testados em 36 instâncias geradas a partir do problema real estudado. Como resultado, percebe-se que as abordagens propostas são melhores que o modelo exato.

O problema estudado por Cappadonna et al (2013) considera máquinas paralelas não relacionadas e limitação de recursos humanos para o processamento da tarefa. Aborda-se a solução por meio de um modelo de programação inteira e por um algoritmo genético (GA). Foram geradas 80 instâncias pequena para testes e comparações com o MILP e os resultados demonstraram eficiência do GA. Também foram geradas 270 instâncias grandes e muito grandes para verificação do comportamento do GA. Percebeu-se que mesmo para instâncias muito grandes, o tempo computacional exigido é baixo. Os resultados do GA, porém, não foram comparados com nenhum outro.

Fanjul et al (2015) consideram o problema de sequenciamento não especificado e dinâmico em máquinas paralelas não relacionadas com recurso adicional (UUPMSR). O problema é não especificado por não ter designação prévia de tarefa-máquina; e dinâmico porque a

alocação de recursos às máquinas não é fixa durante todo o processamento das tarefas. Apresentaram um modelo de programação inteira e um algoritmo de fixação, onde a solução de um UPM serve para fixar a designação tarefa-máquina no UUPMSR, reduzindo a área de busca por solução. Das 582 instâncias resolvidas pelas duas abordagens, as soluções obtidas pelo algoritmo proposto resultaram em *makespan* médio menor que o da programação inteira, devido ao alcance do tempo limite.

Bitar et al (2016) estudam um caso prático de UPM com restrição de recurso com tempo de *setup* dependente da máquina e da sequência, onde o processo de fotolitografia em uma fábrica de semicondutores é um gargalo na produção e o objetivo do problema é maximizar a produção e minimizar o *makespan*. A resolução do problema foi abordada com um *memetic algorithm*.

O estudo de Afzalirad e Rezaeian (2016) aborda diversos aspectos envolvidos no problema de sequenciamento em máquinas paralelas. No modelo proposto são considerados a utilização de recursos renováveis, o tempo de *setup* dependente da sequência e da máquina, a elegibilidade das máquinas, o instante de disponibilidade das tarefas e restrições de precedência entre tarefas. O MILP formulado foi validado com um exemplo numérico considerado pequeno, mas não foi considerado adequado para a resolução de problemas de grande porte, tendo em vista a complexidade e abrangência do modelo. Os autores, então, apresentaram duas abordagens metaheurísticas para a solução do problema, sendo um GA e um Sistema Imunológico Artificial (AIS). Instâncias foram geradas de maneira aleatória e as soluções obtidas pelas metaheurísticas foram comparadas com a solução ótima ou, dependendo do caso, com o *Lower Bound*, tendo em vista que o modelo exato não foi capaz de resolver as instâncias maiores. Os resultados demonstraram que as metaheurísticas foram efetivas e eficientes, mas que o AIS demonstrou uma performance melhor.

Zheng e Wang (2016) consideram em seu estudo um problema de sequenciamento em máquinas paralelas não relacionadas com restrição de recurso (UPMSP_RC), cuja utilização está limitada a uma unidade de recurso por tarefa e sua disponibilidade é menor do que a quantidade de máquinas. Apresentam um modelo de programação inteira e um algoritmo *fruit fly* de duas fases. Na primeira fase, tarefas são designadas para as máquinas cujos tempos de processamento é menor e são ordenadas do maior para o menor tempo de processamento em cada máquina. A solução obtida é utilizada como local central inicial para a busca pelas *fruit flies*, reduzindo assim o espaço de solução. A abordagem de solução demonstrou efetividade e eficiência na solução do UPMSP_RC.

Fanjul et al (2017) propõe o problema de sequenciamento em máquinas paralelas não relacionadas com restrição de recursos renováveis e apresentam dois MILP para a resolução deste. O primeiro deles é uma adaptação do modelo apresentado por Edis e Oguz (2012), que considera que a utilização de recursos adicionais pode tornar menores os tempos de processamento das tarefas (UPMR-S). O segundo e mais importante modelo apresentado (UPMR-P) é baseado no modelo de *strip-packing 2D*, onde fixa-se uma das dimensões da tira (altura que analogamente significa a quantidade de recursos disponíveis) e busca-se a minimização da outra dimensão (largura que corresponde ao *makespan*), tendo, como variáveis, tarefas com dimensões que variam de uma máquina para outra (tanto em uso de recurso renovável quanto em tempo de processamento). Entretanto, este modelo não se mostrou capaz de resolver problemas de instâncias consideradas de médio tamanho. Desta forma, os autores propuseram três Matheurísticas para cada MILP. Como resultado, verificou-se que os MILP obtiveram respostas para os problemas pequenos, mas que os problemas médios somente foram resolvidos pelas Matheurísticas (as aplicadas sobre o UPMR-P tiveram

melhores resultados do que as aplicadas sobre o (UPMR-S). Testes iniciais com instância grandes demonstraram que estudos precisam ser desenvolvidos para que se obtenham boas soluções para instâncias maiores.

Villa et al (2018) se concentram em apresentar heurísticas mais eficientes e efetivas do que as apresentadas por Fanjul et al (2017) por meio de duas abordagens. Uma abordagem baseada nos recursos que consiste de três fases: (a) ordenação das tarefas (desconsiderando a utilização dos recursos); (b) construção da solução (conferindo a factibilidade); e (c) melhoramento da solução por meio de uma Busca Local. Uma segunda abordagem que é baseada na designação tarefa-máquina, consiste de duas fases: (a) tem-se a construção de 8 soluções iniciais distintas, a verificação de factibilidade das mesmas e a correção por um mecanismo de reparo (caso infactível); e (b) após a fase de construção é aplicada uma heurística de multi-fases em todas as soluções encontradas, utilizando combinações de métodos, como buscas locais (considerando e desconsiderando os recursos), ao fim desta aplicação é selecionada a melhor solução dentre as 8 encontradas.

Zheng e Wang (2018) propõe um novo problema, cujo objetivo é minimizar o makespan e o total de emissões de carbono (*resource constraint unrelated parallel machine green scheduling problem* – RCUPMGSP). Novamente utilizam uma abordagem heurística *fruit fly*.

Afzalirad e Shafipour (2018) utilizam um MILP e duas abordagens de GA para a resolução do UPMR com elegibilidade de máquinas.

Fleszar e Hindi (2018) propõe um MILP para a resolução do UPMR com duas máquinas e, para problemas com mais máquinas, desenvolvem uma abordagem heurística de dois estágios, em que primeiramente um MILP calcula um *Lower Bound* e designa tarefas para máquinas e depois um CP finaliza a solução.

Este trabalho, apresenta um modelo de programação linear inteira mista para o UPMR, considerando a utilização de recursos renováveis no processamento das tarefas, considerando também, o tempo de *setup* e de disponibilidade das tarefas e a elegibilidade das mesmas para as máquinas. O trabalho também propõe a utilização de cálculos de *Lower Bound* e *Upper Bound*, visando restringir o espaço de busca e auxiliar na convergência do modelo apresentado.

3. Definição do Problema e Modelo Matemático

O Problema de Sequenciamento em Máquinas Paralelas não Relacionadas (UPM) consiste na ordenação de um conjunto de tarefas em um conjunto de máquinas dispostas em paralelo, onde as tarefas podem ser processadas por qualquer uma das máquinas e o tempo de processamento das tarefas pode variar de uma máquina para outra. Normalmente o objetivo deste problema é a minimização do *makespan*, ou seja, terminar a última tarefa do conjunto no menor instante de tempo possível. Utilizando programação linear inteira mista é possível solucionar o UPM clássico. Tomando X_{im} , como uma variável binária, a qual indica se a tarefa i foi designada à máquina m ($X_{im} = 1$). Também, considerando a variável C_{max} como sendo o *makespan*, o instante de tempo em que a última tarefa é finalizada. O modelo clássico do UPM é dado como:

$$\text{Min } C_{max} \quad (1)$$

$$\text{S.a. } \sum_{m=1}^M X_{im} = 1 \quad \forall i; \quad (2)$$

$$\sum_{i=1}^I P_{im} \times X_{im} \leq C_{\max} \quad \forall m; \quad (3)$$

$$X_{im} \in \mathbb{B} \quad \forall i; \forall m; \quad (4)$$

A função objetivo (1) minimiza o *makespan*. O conjunto de restrições (2) impõe que cada trabalho somente pode ser realizado em uma única máquina. O conjunto de restrições (3) realizam o cálculo do *makespan*, garantindo que seja ao menos o tempo de processamento total de cada máquina P_{im} . Garantem que cada trabalho seja precedido por apenas um outro trabalho na mesma máquina. A restrição (4) representa o domínio das variáveis de decisão.

O Problema de Sequenciamento não Especificado e Dinâmico em Máquinas Paralelas não Relacionadas, com restrição de Recursos Renováveis (UPMR) considera a utilização de recursos renováveis no processamento das tarefas que estão disponíveis em quantidade limitada e que, após sua utilização no processamento de uma tarefa, podem ser reutilizados. Além disso, o problema também considera a elegibilidade das máquinas realizarem determinadas tarefas. Também são considerados os tempos de *setup* e os instantes de disponibilidade das tarefas. O tempo de *setup* é o tempo de preparo da máquina entre duas tarefas distintas. Já o instante de disponibilidade é o instante no qual a tarefa está disponível para ser processada se houver máquina e recurso necessários. O modelo é definido por:

Índices	Descrição
$i, j = 1 \dots J$	Conjunto das tarefas;
$m = 1 \dots M$	Conjunto das máquinas;
$t = 1 \dots T$	Conjunto dos instantes de tempo no horizonte de tempo T ;

Fonte: Os Autores

Tabela 1 – Conjuntos do modelo matemático

Parâmetros	Descrição
P_{im}	Tempo de processamento da tarefa i na máquina m ;
R_i	Instante de disponibilidade da tarefa i ;
S_{ijm}	Tempo de <i>setup</i> alternando da tarefa i para a tarefa j na máquina m ;
Eg_{im}	1, se a tarefa i pode ser realizada na máquina m ; 0, caso contrário;
Res_{im}	Quantidade de recurso necessário para realizar a tarefa i na máquina m ;
Res_{limit}	Limite de recurso disponível para a realização das tarefas;
L	Número suficientemente grande;

Fonte: Os Autores

Tabela 2 – Parâmetros do modelo matemático

Variáveis	Tipo	Descrição
$C_{\max}$	$\mathbb{Z}$	<i>Makespan</i> máximo;
X_{im}	$\mathbb{B}$	=1, se a tarefa i foi realizada na máquina m ; = 0, c.c.;
W_{imt}	$\mathbb{B}$	=1, se a tarefa i foi finalizada na máquina m no instante t ; = 0, c.c.;
Y_{ijm}	$\mathbb{B}$	=1, se a tarefa i é precedente da tarefa j na máquina m ; = 0, c.c.;
Z_{it}	$\mathbb{B}$	=1, se a tarefa i foi finalizada no instante de tempo t ; = 0, c.c.;

Fonte: Os Autores

Tabela 3 – Variáveis do modelo matemático

Modelo:

$$\text{Min } C_{max} \quad (5)$$

$$S.a. \quad \sum_{m=1}^M X_{im} = 1 \quad \forall i = 1, \dots, J; \quad (6)$$

$$\sum_{(i=0, i \neq j)}^J Y_{ijm} = X_{jm} \quad \forall m; \forall j = 1, \dots, J; \quad (7)$$

$$\sum_{(i=0, i \neq j)}^J Y_{ijm} \leq X_{im} \quad \forall m; \forall i = 1 \dots J; \quad (8)$$

$$\sum_{j=1}^J Y_{0jm} \leq 1 \quad \forall m; \forall j = 1 \dots J; \quad (9)$$

$$X_{im} \leq E g_{im} \quad \forall m; \forall i = 1 \dots J; \quad (10)$$

$$\sum_{t=1}^T Z_{it} = 1 \quad \forall i = 1 \dots J; \quad (11)$$

$$\sum_{t=P_{jm}}^T t \times Z_{jt} + L(1 - Y_{0jm}) \geq P_{jm} + S_{0jm} \quad \forall m; \forall j = 1 \dots J; \quad (12)$$

$$\sum_{t=P_{jm}}^T t \times Z_{jt} - \sum_{t=P_{im}}^T t \times Z_{it} + L(1 - Y_{ijm}) \geq P_{jm} + S_{ijm} \quad \forall m; \forall i = 0, \dots, J; \forall i \neq j = 1, \dots, J; \quad (13)$$

$$\sum_{t=P_{im}}^T t \times Z_{jt} \geq R_j + (S_{ijm} + P_{jm}) \times Y_{ijm} \quad \forall m; \forall i = 0, \dots, J; \forall i \neq j = 1, \dots, J; \quad (14)$$

$$\sum_{i=1}^J \sum_{m=1}^M \sum_{s=t}^{t+P_{im}-1} Res_{im} \times W_{imt} \leq Res_{limit} \quad \forall t = 1, \dots, T; \quad (15)$$

$$\sum_{t=1}^T W_{imt} \leq X_{im} \quad \forall m; \forall i = 1, \dots, J; \quad (16)$$

$$\sum_m^M W_{imt} \leq Z_{it}; \quad \forall m; \forall t = 1, \dots, T; \quad (17)$$

$$W_{imt} \geq X_{im} + Z_{it} - 1 \quad \forall m; \forall i = 1 \dots J; \forall t = 1, \dots, T; \quad (18)$$

$$\sum_{t=1}^T t \times Z_{it} \leq C_{max} \quad \forall i = 1 \dots J; \quad (19)$$

$$LB \leq C_{max} \quad \forall i = 1 \dots J; \quad (20)$$

$$\sum_{i=1}^J \sum_{t=1}^{\min\{P_{im}\}-1} \sum_{m=1}^M Z_{it} \times W_{imt} = 0 \quad (21)$$

$$\sum_{t=1}^{P_{im}} Z_{it} + W_{imt} \leq 1 - X_{im} \quad \forall m; \forall i = 1 \dots J; \quad (22)$$

$$C_{max} \in \mathbb{Z} \quad (23)$$

$$X_{im}, Y_{ijm}, W_{imt}, Z_{it} \in \mathbb{B} \quad \forall m; \forall i; \forall j; \forall t; \quad (24)$$

A função objetivo (5) minimiza o *makespan*. A expressão (6) impõe que cada tarefa somente pode ser realizada em uma única máquina. O conjunto de restrições (7) e (8) garante que cada tarefa seja precedida por apenas uma outra tarefa na mesma máquina. A restrição (9) garante que a tarefa artificial 0 esteja posicionada no início de todas as máquinas, podendo ou não serem utilizadas. A expressão (10) garante que a tarefa i , somente pode ser realizada na máquina m , se a mesma for elegível para a máquina. A expressão (11) garante que as tarefas sejam finalizadas apenas em um instante de tempo t . As restrições (12) e (13) indicam que o tempo de finalização de uma tarefa seja ao menos igual à soma de seu tempo de *setup*, seu tempo de processamento e do tempo de finalização da tarefa precedente. A expressão (14) garante que o tempo de finalização de cada tarefa, em sua respectiva máquina designada, não é menor que a soma do seu tempo de processamento, seu instante de disponibilidade e o tempo de *setup* relativo ao seu precedente. Além disso, a restrição (14) garante que o *setup* somente se inicie após a liberação da máquina para a tarefa. O conjunto de restrições (15) garante que o limite de recursos total não seja ultrapassado em cada instante de tempo. O conjunto de restrições (16)-(18) garante que uma tarefa será finalizada em uma máquina em um determinado instante de tempo ($W_{imt} = 1$) somente se for designada para esta máquina ($X_{im} = 1$) e se for finalizado neste determinado instante de tempo ($Z_{it} = 1$). A expressão (19) calcula o *makespan*. A restrição (20) garante que o *Lower Bound* seja respeitado. Os conjuntos de restrições (21) e (22) implicam que o tempo mínimo de finalização das tarefas, a partir do início do horizonte de tempo, seja seu tempo de processamento. Além disso, os conjuntos de restrições (21) e (22) são inequações válidas que auxiliam na convergência do modelo. As restrições (23) e (24) representam o domínio das variáveis de decisão.

4. Definição de Lower Bounds e Upper Bounds

Dado que o Problema de Sequenciamento de Máquinas Paralelas não Relacionadas com restrição de Recursos Renováveis é considerado *NP-Hard*, esta pesquisa utiliza de artifícios, como a definição de *Lower e Upper Bounds* para restringir a área de busca e facilitar a resolução do modelo, auxiliando na sua convergência. Os *Lower Bounds* utilizados foram propostos anteriormente para outras variantes deste problema (Fanjul et al, 2017; Grigoriev et al, 2005), e são calculados baseando-se no problema clássico de máquinas paralelas:

$$LB_1 = \max_i \min_m P_{im},$$

$$LB_2 = \left\lceil \frac{1}{m} \sum_i \min_m P_{im} \right\rceil,$$

O primeiro *Lower Bound* (LB_1) é calculado a partir do princípio que o *makespan* deve ser maior ou igual ao tempo de processamento mínimo de cada trabalho. Já o segundo *Lower*

Bound (LB_2) é calculado a partir da divisão entre a soma dos mínimos tempos de processamento de cada trabalho e o número de máquinas.

Foi observado por Fanjul et al (2017), que o terceiro *Lower Bound* (LB_3) pode ser obtido ao resolver o modelo clássico de máquinas paralelas, apresentado em (1)-(4). Com isso, para evitar buscas desnecessárias pelo método de *Branch-and-Bound* e auxiliar na velocidade de convergência do modelo proposto por esta pesquisa, o *Lower Bound* foi definido como:

$$\text{Lower Bound} = \max\{LB_1, LB_2, LB_3\}$$

O cálculo do *Upper Bound* se faz necessário devido ao conjunto T , conjunto dos instantes de tempo no horizonte de tempo, o qual é responsável por determinar a quantidade de variáveis Z_{it} e W_{imt} a serem criadas, pois quanto menor o horizonte, menor a quantidade de variáveis utilizadas, logo, maior a facilidade de convergência do modelo. Para isto, foi desenvolvida uma heurística construtiva, afim de encontrar uma solução mínima possível e factível para o problema, assim limitando superiormente o espaço de busca. A heurística é apresentada na Figura 1.

Algoritmo 1: ALGORITMO PARA DEFINIÇÃO DO HORIZONTE DE TEMPO PARA $m = 2$

```

1 //Variáveis auxiliares:
2 int Maquina_Atual; //Variável que indica a máquina que está selecionada;
3 int Outra_Maquina; //Variável que indica a máquina que não está selecionada;
4 bool Troca; //Variável que indica se houve troca de máquinas
5 para m = 1 até |M| faça
6   Ordenar os trabalhos por Recurso (decrescente) e Tempo (crescente);
7   T[m]=0; //Variável que indica o tempo de processamento total da máquina m;
8   R[m]=0; //Variável que indica o recurso utilizado pela máquina m;
9 fim
10 Criar uma lista de tarefas não atribuídas com todos as tarefas;
11 Selecionar a tarefa i com menor recurso e a respectiva máquina m;
12 Retirar a tarefa i selecionada da lista de não atribuídas;
13 Definir Maquina_Atual, como a máquina da tarefa selecionada;
14 Definir Outra_Maquina, como a máquina não selecionada;
15 T[Maquina_Atual] = Processamento[i, Maquina_Atual] //Atualizar o tempo da máquina selecionada;
16 R[Maquina_Atual] = Recurso[i, Maquina_Atual] //Atualizar o recurso da máquina selecionada;
17 enquanto (∃ Tarefa não atribuída) faça
18   Troca = Falso;
19   para i = 1 até |I| faça
20     se ((Rlimit - R[Maquina_Atual] ≥ Recurso[i, Outra_Maquina]) e (i ∈ Lista) então
21       T[Outra_Maquina] += Processamento[i, Outra_Maquina];
22       R[Outra_Maquina] = Recurso[i, Outra_Maquina];
23       se (T[Maquina_Atual] < T[Outra_Maquina]) então
24         Troca = Verdadeiro;
25         Realizar a troca de máquinas;
26       Pare;
27     fim
28   fim
29 fim
30 se (Troca == Falso) então
31   T[Outra_Maquina] = T[Maquina_Atual];
32   R[Outra_Maquina] = 0;
33   Selecionar a tarefa i não atribuída com menor recurso máquina Maquina_Atual;
34   Retirar a tarefa i selecionada da lista de não atribuídas;
35   T[Maquina_Atual] = Processamento[i, Maquina_Atual]
36   R[Maquina_Atual] = Recurso[i, Maquina_Atual]
37 fim
38 fim
39 retorna Max{T[Maquina_Atual], T[Outra_Maquina]};

```

Figura 1 - Algoritmo para definição do Horizonte de Tempo para $m = 2$

Os passos de 1-9 inicializam as variáveis. O passo 10 é responsável por criar uma lista com todas as tarefas não atribuídas. Nos passos 11-12 é selecionada a tarefa de menor recurso e

sua respectiva máquina e realizada a remoção da tarefa da lista de não atribuídas. Os passos 13-16 atribuem valores às variáveis responsáveis pela máquina que está sendo analisada, seu respectivo tempo e recurso atual. O loop 17-38 é realizado enquanto todas as tarefas não forem atribuídas. O passo 18 atribui à variável *Troca* o valor de falso, indicando que no início do processo não houve troca de máquinas, ou seja, ainda está sendo analisada a mesma máquina. O loop 19-29 é responsável por verificar todas as tarefas. O passo 20 é uma condição para verificar se a subtração do recurso limite e do recurso da máquina que está sendo avaliada é maior que o recurso da tarefa verificada na outra máquina e se esta tarefa ainda não foi atribuída. Caso seja atribuída, os passos 21-22 realizam a atualização do tempo e do recurso desta segunda máquina. O passo 23 é uma condicional, para verificar se o tempo da máquina que está sendo verificada foi ultrapassado pela outra máquina, caso sim, atualiza a variável *Troca*, a outra máquina passa ser analisada e é finalizado o loop 19-29. Já o passo 30 é uma condição para ver se foi realizada a troca de máquinas. Caso positivo, os passos 31-32 realizam as atualizações de tempo e recurso da outra máquina. Os passos 33-34 são análogos aos passos 11-12. Enquanto os passos 35-36 são análogos aos passos 15-16. Por fim, o passo 39 retorna o maior tempo entre as duas máquinas.

5. Resultados computacionais

O objetivo deste capítulo é analisar a performance do modelo proposto, avaliando a eficiência de ambos os métodos, exato e heurístico. Para realizar a implementação computacional foi utilizado a linguagem de programação C# e utilizado o solver Gurobi 8.0.0 para solucionar o problema. O computador utilizado possui um processador Intel Core i5, 8 Gb de memória RAM.

Os dados utilizados para validação do modelo foram retirados do exemplo originalmente apresentado por Fanjul et al (2017). Considere o seguinte exemplo, são definidas 2 máquinas para 5 tarefas, com limite máximo de recurso (Res_{limit}) igual a 5. Considere também os seguintes tempos de processamento e recursos necessários:

$$P_{im} = \begin{pmatrix} 1 & 2 & 2 & 2 & 1 \\ 2 & 1 & 2 & 3 & 1 \end{pmatrix} \quad Res_{im} = \begin{pmatrix} 4 & 3 & 3 & 4 & 2 \\ 2 & 5 & 4 & 2 & 5 \end{pmatrix}$$

O exemplo apresentado não considera os tempos de *setup* e instante de disponibilidade, nem a elegibilidade das máquinas. Para adaptar para o modelo proposto, os valores de tempo de *setup* foram zerados e as tarefas foram consideradas elegíveis para todas as máquinas. Com estes dados foi obtida a solução ótima de $C_{max} = 5$, da mesma forma que apresentada originalmente.

Para realizar os testes computacionais foram utilizadas 22 das instâncias apresentadas por Fanjul et al. (2017), 14 destas instâncias consideram 8 tarefas a serem realizadas, enquanto as outras 8 instâncias consideram 12 tarefas. O tempo limite estabelecido, para os testes, foi de 3600 segundos. Foram realizados dois testes nestas instâncias, o primeiro foi sem o cálculo do *Upper Bound*. Já o segundo foi utilizando a heurística construtiva proposta por esta pesquisa. A comparação dos resultados obtidos com os resultados de Fanjul et al. (2017), para os testes sem e com utilização da heurística construtiva para o cálculo do *Upper Bound*, são apresentados na Tabela 4 e 5, respectivamente.

Para os testes sem a utilização da heurística construtiva para o cálculo do *Upper Bound*, foi considerado o horizonte de tempo como $\max\{\sum_i^J P_{i1}, \sum_i^J P_{i2}\}$. No entanto, como pode ser observado na Tabela 4, o tempo limite de 3600 segundos foi atingido em todas as 22 instâncias e não foi possível encontrar nenhuma solução incumbente.

Quantidade de Tarefas	Resultados	Quantidade de Instâncias		Gap(%)
		Tempo Computacional	Solução	
8 Tarefas	Melhor	-	-	-
	Igual	-	-	-
	Inferior	14	-	-
12 Tarefas	Melhor	-	-	-
	Igual	-	-	-
	Inferior	8	-	-

Fonte: Os Autores

Tabela 4 – Resultados Computacionais para testes sem cálculo do *Upper Bound*.

Já os resultados da Tabela 5, com a utilização da heurística construtiva para o cálculo do *Upper Bound*, demonstram que foi possível encontrar a solução ótima e igual ao dos autores em 12 dos testes realizados (8 instâncias de 8 tarefas e 4 instâncias de 12 tarefas). Destas soluções, foi possível melhorar o tempo em comparação com os autores em 6 instâncias, foi obtido um tempo similar em 4 instâncias e um tempo inferior em 2 instâncias. Nos 10 testes em que foi obtida solução inferior, o tempo limite estabelecido de 3600 segundos foi atingido. No entanto, o *gap* obtido nas soluções foi em média de 3,45% para as 6 instâncias de 8 tarefas e de 3,54% para as 4 instâncias de 12 tarefas.

Quantidade de Tarefas	Resultados	Quantidade de Instâncias		Gap(%)
		Tempo Computacional	Solução	
8 Tarefas	Melhor	4	-	-
	Igual	2	8	0,00%
	Inferior	8	6	3,45%
12 Tarefas	Melhor	4	-	-
	Igual	2	8	0,00%
	Inferior	8	6	3,45%

Fonte: Os Autores

Tabela 5 – Resultados Computacionais para testes com cálculo do *Upper Bound*.

5. Considerações Finais

Neste artigo foi apresentado um modelo matemático de programação linear inteira mista para o Problema de Sequenciamento não Especificado e Dinâmico em Máquinas Paralelas não Relacionadas, com restrição de Recursos Renováveis (UPMR), também foram apresentadas abordagens para restringir o espaço de busca, como o cálculo de *Lower* e *Upper Bounds*, visando auxiliar na velocidade de convergência.

Com este trabalho foi possível expor a dificuldade em solucionar problemas mais próximos à realidade utilizando apenas modelos de programação linear inteira mista. Evidenciando assim, a necessidade de utilizar outros métodos de solução, como heurísticas e/ou metaheurísticas.

Para futuros estudos, serão considerados problemas maiores e mais próximos à realidade, com números maiores de máquinas ($m > 2$) e tarefas ($i > 12$); desenvolvidas heurísticas e metaheurísticas; e a utilização de outras abordagens para restringir o espaço de busca.

Referências

AFZALIRAD, M.; REZAEIAN, J. Resource-constrained unrelated parallel machine scheduling problem with sequence dependent setup times, precedence constraints and machine eligibility restrictions. Computers & Industrial Engineering. Vol 98, p. 40-52, 2016.

AFZALIRAD, M.; SHAFIPOUR, M. *Design of an efficient genetic algorithm for resource-constraint unrelated parallel machine scheduling problem with machine eligibility restrictions.* Journal of Intelligent Manufacturing, Vol. 29, p. 423-437, 2018.

BITAR, A.; DAUZÈRE-PÉRÈS, S.; YUGMA, C.; ROUSSEL, R. *A memetic algorithm to solve an unrelated parallel machine scheduling problem with auxiliary resources in semiconductor manufacturing.* Journal of Scheduling, Vol. 19, p. 367-376, 2016.

CAPPADONNA, F.A.; COSTA, A.; FICHERA, S. *Makespan minimization of unrelated parallel machine with limited human resources.* Conference on Intelligent Computation in Manufacturing Engineering (CIRP). 8^a ed, Nápoles, Itália, p. 450-455, 2013.

EDIS, E.B.; OZKARAHAN, I. *A combined integer/constraint programming approach to a resource-constrained parallel machine scheduling problem with machine eligibility restrictions.* Engineering Optimization, Vol 43, p. 135-157, 2011a.

EDIS, E.B.; OZKARAHAN, I. *Solution approaches for a real-life resource-constrained parallel machine scheduling problem.* The International Journal of Advanced Manufacturing Technology, Vol. 58, p. 1141-1153, 2011b.

FANJUL, L.P.; PEREA, F.; RUIZ, R. *Algorithms for the Unspecified Parallel Machine Scheduling Problem with additional Resource.* International Conference of Industrial Engineering and Systems Management (IESM) 6^a ed. Sevilha, Espanha, 2015.

FANJUL, L.P.; PEREA, F.; RUIZ, R. *Models and matheuristics for the unrelated parallel machine scheduling problem with additional resources.* European Journal of Operational Research, Vol. 260, p. 482-493, 2017.

FLESZAR, K.; HINDI, K.S. *Algorithms for the unrelated parallel machine scheduling problem with a resource constraint.* European Journal of Operational Research, Vol. 271, p. 839-848, 2018.

GRIGORIEV, A.; SVIRIDENKO, M.; UETZ, M. *Unrelated parallel machine scheduling problem with resource dependent processing times.* Conference of Integer Programming and Combinatorial Optimization (IPCO). 11^a ed. Berlin, Alemanha, 2005.

KAMPKE, E.H.; ARROYO, J.E.C.; SANTOS, A.G. *Reactive GRASP with path relinking for solving parallel machine scheduling problem with resource-assignable sequence dependent setup times.* World Congress on Nature & Biologically Inspired Computing (NaBIC 2009), Coimbatore, India, p. 924-929, 2009.

KAMPKE, E.H.; ARROYO, J.E.C.; SANTOS, A.G. *Iterated Local Search with Path Relinking for solving parallel machine scheduling problem with resource-assignable sequence dependent setup times.* European Conference on Evolutionary Computation in Combinatorial Optimization. 10^a ed. Istanbul, Turquia, p. 107-118, 2010.

RUIZ, R.; ANDRÉS, C. *Unrelated parallel machines scheduling with resource-assignable sequence dependent setup times.* Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA). 3^a ed, Paris, França, p. 439-446, 2007.

VENTURA, J.A.; KIM, D. *Parallel machine scheduling about an unrestricted due date and additional resource constraints.* IIE Transactions, Vol. 32, p. 147-153, 2000.

VENTURA, J.A.; KIM, D. *Parallel machine scheduling with earliness-tardiness penalties and additional resource constraint.* Computers & Operations Research, Vol 30, p. 1945-1958, 2003.

VILA, F.; VALLADA, E.; FANJUL, L.P. *Heuristic algorithms for the unrelated parallel machine scheduling problem with one scarce additional resource.* Expert Systems with Applications, Vol 93, p. 28-38, 2018

ZHENG, X.-I.; WANG, L. *A two-stage adaptative fruit fly optimization algorithm for unrelated parallel machine scheduling problem with additional resource constraint.* Expert Systems with Applications, Vol. 65, p. 29-39, 2016.

ZHENG, X.-I.; WANG, L. *A collaborative Multiobjective Fruit Fly Optimization algorithm for the resource constraint unrelated parallel machine green scheduling problem.* IEEE Transactions on Systems, Man and Cybernetics: Systems, Vol. 48, p. 790-800, 2018.