

HEURÍSTICAS E ADEQUAÇÕES DO MODELO MATEMÁTICO APLICADOS AO PROBLEMA DE PROGRAMAÇÃO DE TAREFAS EM MÚLTIPLAS FÁBRICAS COM MÁQUINAS PARALELAS RELACIONADAS

Felipe Martins Muller

Universidade Federal de Santa Maria - Centro de tecnologia - UFSM/CT
Programa de Pós-Graduação em Engenharia de Produção - PPGE
Departamento de Computação Aplicada - DCOM
Av. Roraima nº 1000, Cidade Universitária, Bairro Camobi, Santa Maria - RS
felipe@inf.ufsm.br

Angelita Pezzi Pasqualon Bridi *

angelitabridi@gmail.com

Scheila Daiana Severo Hollveg *

scheilahollveg@gmail.com

Suzianne Miranda Tormann *

suzianne.tormann@redes.ufsm.br

* Universidade Federal de Santa Maria - UFSM

Programa de Pós-Graduação em Engenharia de Produção - PPGE
Av. Roraima nº 1000, Cidade Universitária, Bairro Camobi, Santa Maria - RS

RESUMO

O sequenciamento da produção é uma necessidade cotidiana observada nas plantas fabris. O presente estudo trata do problema de programação de tarefas em múltiplas fábricas com máquinas paralelas relacionadas, propondo uma comparação entre um Limitante Inferior, um Algoritmo Genético com busca local (GALS) e uma Busca Tabu (BT), inicialmente para a minimizar o (*Makespan*). Também é apresentada uma adequação da modelo matemático para a minimização do tempo total do término das tarefas (*Total Completion Time*) e as soluções obtidas pelo CPLEX foram comparadas com o GALS. Foram analisados 3 cenários em um conjunto de 30 diferentes instâncias. Os resultados obtidos apresentaram-se bastante competitivos.

PALAVRAS CHAVE. Máquinas Paralelas Relacionadas, Metaheurísticas, Múltiplas-Fábricas.

Tópicos: MH - Metaheurísticas, OA – Outras Aplicações em PO

ABSTRACT

The sequencing and scheduling of tasks is a daily necessity observed in plant production. This research deals with the problem of programming tasks in multiple factories with related parallel machines, proposing a comparison between a lower bound, a Genetic Algorithm with local search and a Tabu Search, initially to minimize the Makespan. Also presented is the adequacy of the mathematical model for minimizing the total time of completion of tasks and a comparison with GALS were performed. Three scenarios were analyzed in a set of 30 different instances. The results were very competitive.

KEYWORDS. Related Parallel Machine. Metaheuristics. Multiple-Factories.

Paper topics: MH – Metaheuristics, OA – Other Applications of OR

1. Introdução

Problemas de sequenciamento de produção (*sequencing and scheduling*), são bastante atrativos, uma vez que os desafios a serem resolvidos, tornam-se mais complexos com o passar do tempo e os avanços tecnológicos. O uso de ferramentas que tragam melhorias a solução do problema diversificam-se com o decorrer dos anos, buscando otimização e aprimoramento dos processos com intuito de minimizar os diversos objetivos relacionados a este problema. Neste artigo iremos tratar da minimização do *Makespan* e do tempo total de finalização das tarefas.

São muitas as preocupações dos responsáveis pelo setor de produção, seja, gerente, diretor ou até mesmo um administrador de alguma indústria, que muitas vezes não possuem ferramentas teóricas para auxiliarem em sua logística [da Silva Xavier et al., 2020]. Além do mais, as consequências estratégicas de escalonamentos pobres no desempenho global da empresa não são analisadas e tais resultados são mencionados como: processos mais demorados que afetam na finalização dos produtos, gerando atraso nas entregas, aumento de horas extras e descontentamento do cliente devido o atraso no recebimento de sua compra. Isso acaba causando uma imagem negativa para a empresa ocasionando diversas perdas [Fuchigami e Rangel, 2014].

Por isso, decisões estão se tornando mais difíceis, principalmente, com a evolução no meio industrial e as condições impostas do mercado atual. Desse modo, um processo de tomada de decisão eficaz constitui-se essencialmente de algumas fases: a formulação do problema, a construção de um modelo matemático, a sua resolução precisa ser realizada e por fim, a implementação da solução obtida e eventuais correções [Chaouch et al., 2020].

Sendo assim, o delineamento do presente estudo baseou-se nos resultados obtidos por [Behnamian e Fatemi Ghomi, 2013], ao passo que buscou confrontar duas meta-heurísticas um AG e uma BT, analisando 3 cenários contendo as instâncias do problema em questão. Objetivou-se minimizar o *Makespan* e apresentar uma adequação da modelagem matemática para o tempo total do término das tarefas, *Total Completion Time*, tornando-a mais adequada a solução por um solver comercial.

Este artigo está estruturado da seguinte forma: Na Seção 2 apresenta-se uma Revisão Bibliográfica. Na Seção 3, encontra-se a definição do problema, o modelo matemático e as duas meta-heurísticas propostas para comparação. Na Seção 4, descreve-se o conjunto de instâncias aplicadas. A Seção 5, apresenta os resultados computacionais obtidos. E por fim, na Seção 6 as conclusões.

2. Revisão bibliográfica

O estudo de [Behnamian e Fatemi Ghomi, 2013] utilizou um modelo baseado em programação linear inteira mista relacionado ao problema de máquinas paralelas em multi-fábricas (HMFPS, do inglês, *Heterogeneous Multi-Factory Production network Scheduling*), com o objetivo de minimizar o *Makespan* entre as fábricas. Os autores propuseram um novo método heurístico baseado na regra LPT (do inglês, *Longest Processing Time first*), chamado de MLPT (do inglês, *Modified LPT*) LPT que combina as regras da alocação de tarefas do LPT para decidir em que fábrica e em que máquina elas devem ser executadas. Para o mesmo problema desenvolveram uma metaheurística híbrida combinando um algoritmo genético com busca local, também propuseram uma nova forma de representar a solução no cromossomo para tratar da questão de múltiplas fábricas.

[Yazdani et al., 2015] abordaram o problema de programação de redes de produção em multi-fábricas distribuídas com máquinas paralelas. Para solução inicial do algoritmo construtivo, os pesquisadores utilizaram a regra de despacho SPT (do inglês, *Shortest Processing Time First*), para alocação nas máquinas. Ainda, propuseram três modelos matemáticos, que só conseguiram

solucionar problemas de pequena dimensão. Então, apresentaram um algoritmo de colônias de abelhas artificiais, com o objetivo de minimizar o (*Makespan*) e o *Total Completion Time* (TCT) que foi comparado com o algoritmo genético apresentado por [Behnamian e Fatemi Ghomi, 2013]. Este artigo apresenta um modelo matemático superior aos três apresentados por [Yazdani et al., 2015].

[Joo e Kim, 2017] abordaram o problema de sequenciamento integrado de tarefas em máquinas paralelas não-relacionadas, de lotes e de entregas com frota heterogênea. Eles propuseram duas soluções: uma exata baseada em um modelo de programação linear inteira mista; e uma heurística através de um Algoritmo Genético. Ambos buscaram encontrar a combinação de alocação de tarefas, lotes e frota de modo a minimizar o *Makespan* de todo o processo.

A pesquisa de [Huo, 2019] apresentou um algoritmo de tempo polinomial ótimo para o sequenciamento em máquinas paralelas preemptivo sujeito à disponibilidade das máquina com o objetivo de minimizar o *Makespan*, sujeito a restrição que o *Total Completion Time* seja minimizado. Um apanhado dos problemas de sequenciamento considerando múltiplas fábricas pode ser visto em [Behnamian e Fatemi Ghomi, 2016].

A próxima seção apresenta os problemas abordados e seus modelos matemáticos.

3. Definição do problema

Para formalizar o HMFPS, assume-se um conjunto F de fábricas que vai de $\{1, 2, \dots, f\}$. Em cada fábrica temos o conjunto M , de máquinas idênticas paralelas, ou seja, a máquina M_1 pertencente a fábrica F_1 possui o mesmo tempo de processamento da máquina M_2 da mesma fábrica, porém seu tempo de processamento pode ser diferente da máquina M_1 da fábrica F_2 . Existem n tarefas independentes $\{1, 2, \dots, j\}$ a serem alocadas às máquinas das fábricas.

Tendo em vista que cada tarefa pode ser executada por, no máximo, uma máquina e cada máquina pode processar apenas uma tarefa de cada vez, sem preempção, definem-se os parâmetros e as variáveis envolvidas na descrição e modelagem do problema.

Parâmetros:

f : Índice de fábricas;

k : Índice de máquinas;

j : Índice de tarefas a serem processadas;

v_k : Velocidade de processamento de cada máquina k ;

p_j : Tempo de processamento de cada tarefa j ;

$p_{j,k}$: Tempo de processamento de cada tarefa j na máquina k .

Conjuntos:

J : Conjunto de tarefas a serem alocadas;

F : Conjunto de fábricas;

K_f : Conjunto de máquinas da fábrica f ;

3.1. Modelo matemático

O objetivo é encontrar os valores das variáveis de decisão x_{jfk} que assume o valor 1 se a tarefa j está alocada na máquina k da fábrica f e 0 caso contrário. Apresenta-se um modelo matemático para minimizar o *Makespan* e dois modelos matemáticos para minimizar o tempo total de processamento de todas as tarefas, *Total completion time*, um deles inovador e permitindo que os solvers comerciais estejam aptos a resolver problemas de maior dimensão que se fosse usada as outras modelagens. Inicialmente apresenta-se o modelo matemático com o objetivo de minimizar o *Makespan*:

Modelo 1 - *Makespan*

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

$$\text{sujeito a } \sum_{f \in F} \sum_{k \in K} x_{jfk} = 1 \quad \forall j \in J \quad (2)$$

$$\sum_{f,j \in J} \frac{p_j}{v_k} x_{jfk} \leq C_{max} \quad \forall k \in K \quad (3)$$

$$x_{jfk} \in \{0, 1\} \quad (4)$$

$$C_{max} \geq 0 \quad (5)$$

O modelo possui o objetivo de minimizar o valor do C_{max} como descrito na (1), possuindo como restrições a equação (2) que descreve que uma tarefa j só poderá ser processada apenas em uma máquina k de uma fábrica f e a equação (3) diz que tempo de processamento total das tarefas alocadas em cada máquina não deve exceder o *Makespan*. As equações (4) e (5) apresentam o domínio das variáveis.

Os modelos matemáticos 2 e 3, referem-se a minimização do *Total Completion Time* e apresentam diferenças cruciais para serem resolvidos por solvers comerciais.

Modelo 2 - *Total Completion Time*

$$\text{Min } \sum_{j \in J} C_j \quad (6)$$

$$\text{sujeito a } \sum_{i \in J} \sum_{k \in K} x_{ijk} = 1 \quad \forall j \in J \quad (7)$$

$$\sum_{l \in J: l \leq j} \frac{p_j}{v_k} x_{ijk} - (1 - x_{ijk}) \sum_{l \in J} \frac{p_j}{v_k} \leq C_j \quad (8)$$

$$x_{jfk} \in \{0, 1\} \quad (9)$$

$$C_j \geq 0 \quad (10)$$

$$(8), (9) : \forall j \in J, \forall f \in F, \forall k \in K \quad (11)$$

Modelo 3 - *Total Completion Time*

$$\text{Min } \sum_{j \in J} \sum_{f \in F} \sum_{k \in K} \frac{p_j}{v_k} y_{jfk} \quad (12)$$

$$\text{sujeito a } \sum_{f \in F} \sum_{k \in K} x_{jfk} = 1 \quad \forall j \in J \quad (13)$$

$$\sum_{l \in J: l \geq j} x_{lfk} + (x_{jfk} - 1) \cdot (|J|) \leq y_{jfk} \quad (14)$$

$$x_{jfk} \in \{0, 1\} \quad (15)$$

$$y_{jfk} \geq 0 \quad (16)$$

$$(14), (15), (16) : \forall j \in J, \forall f \in F, \forall k \in K \quad (17)$$

No modelo 2, a função (6) tem por objetivo minimizar o somatório de C_j , que é a soma dos tempos de finalização de cada tarefa. Sujeito as restrições (7), no qual estipula que uma tarefa só pode ser processada por apenas em uma máquina k de uma fábrica f . A (8) o tempo de processamento de uma tarefa j pela velocidade de processamento de uma máquina k se alocado na fábrica

f esta sujeito a demais tarefas alocadas na mesma máquina, verifica-se se o somatório do tempo de todas as tarefas a serem processadas são menores ao C_j existente. As equações (9) e (10) estipulam os domínios das variáveis.

No modelo 3, a função (12) tem como objetivo minimizar o valor do processamento total de máquinas, levando em consideração a divisão entre o tempo de processamento de uma tarefa j pela velocidade de processamento da máquina k , a variável y_{jfk} refere-se a alocação da tarefa j em uma máquina k de uma fábrica f .

Esta função objetivo está sujeita a restrições de acordo com as equações (13) e (14), sendo a primeira estipulando que uma tarefa será processada apenas em uma máquina dentro de uma fábrica e a segunda verifica quantas tarefas estão alocadas antes da tarefa a ser processada. As equações (15) e (16) estipulam os domínios das variáveis. Os modelos matemáticos serão usados para serem resolvidos através de solvers comerciais, porém como o HMFPS é NP-difícil, vai-se apresentar dois procedimentos heurísticos para resolvê-los.

3.2. Heurísticas propostas

Nesta seção são detalhadas a Busca Tabu e o Algoritmo Genético utilizados para resolver o problema de minimização do *Makespan*.

3.2.1. Busca tabu

"A Busca Tabu, surgiu no final da década de 1980, originalmente proposta por [Glover, 1989], o método parte de uma solução inicial, normalmente, gerada por uma heurística construtiva e guia um processo de busca local ou busca de vizinhança, buscando o melhor movimento a cada interação". Para impedir que o método retorne a algum ótimo local já visitado, atributos das soluções visitadas anteriormente são armazenados em uma lista tabu e soluções contendo esses atributos são proibidas, a não ser que passem por um critério de aspiração.

Para este estudo, a Busca Tabu foi aplicada com finalidade de redução do *Makespan* da máquina mais carregada, de acordo com o Algoritmo 1:

Algoritmo 1: Algoritmo 1: Busca Tabu aplicado ao critério <i>Makespan</i>	
1	Entrada(arquivo instancias);
2	Solucao Inicial $\leftarrow$ Algoritmo Construtivo;
3	repita
4	ListaTabu $\leftarrow T = n/M$
5	Exploracao de Vizinhanças $\leftarrow$ AreaTransferencia – Permutacao – Cmin
6	Processamento Fluxo do Algoritmo
7	Modo de Melhoria Forçada
8	Modo Piora
9	até critério de parada alcançado;
10	retorna Solucao Incumbente

Para implementação do algoritmo construtivo foram primeiramente ordenadas as tarefas de maneira decrescente. Logo após, as tarefas foram alocadas de maneira dinâmica iniciando pelas máquinas com maiores índices de velocidades. A partir do momento em que todas máquinas tiverem uma tarefa alocada, é realizada a alocação das tarefas seguindo o critério de atribuição para a primeira máquina disponível, ou seja, a máquina que tiver o menor *Makespan* irá receber a maior tarefa a ser alocada.

A lista tabu foi construída como três vetores (um para cada vizinhança) contendo $T=n/m$ elementos. Cada elemento na lista tabu contém o índice das máquinas envolvidas no movimento, assim como o índice das tarefas processadas.

A exploração das vizinhanças foi subdividida em três espaços de vizinhanças, seguindo critérios e procedimentos para permanência ou migração para próxima vizinhança. A primeira vizinhança (1), Área de Transferência, retira uma tarefa aleatória da máquina mais carregada e a realoca na menos carregada, se isso provocar melhoria no *Makespan*. A segunda vizinhança (2), Permutação, tenta trocar uma tarefa da máquina mais carregada com uma aleatória da menos carregada que melhore o *Makespan*. A terceira vizinhança (3), Minimização do *Cmin* (tempo de processamento da máquina menos carregada), tenta trocar uma tarefa entre a máquina menos carregada e alguma outra que não seja a máquina mais carregada, no sentido de minimizar o valor de *Cmin*, abrindo espaço para as outras vizinhanças atuarem.

O modo, fluxo do algoritmo, inicia o processo pela vizinhança 1, a cada $K=0,4(n/M)$ iterações sem melhora na função objetivo o algoritmo entra em modo de melhora forçada, utilizando a vizinhança corrente. Caso o modo de melhora forçada falhar ao produzir uma melhora na função objetivo, aplica-se o rotacionamento da vizinhança. As vizinhanças 1 e 2 são exploradas exaustivamente, porém a vizinhança 3, é executada apenas uma vez antes de ser alternada.

Caso ocorram $L=2(n/M)$ tentativas sem melhora da função objetivo, o algoritmo entra em modo de piora. Neste momento, permite-se um movimento que cause uma piora no *Makespan*, reiniciando-se a exploração das vizinhanças. O algoritmo é executado por 4 ciclos de 200 iterações sem melhora, sendo reiniciado com a solução inicial original, mas mantendo-se a solução incumbente. Isso é usado devido ao caráter aleatório das vizinhanças 1 e 2, que a cada reinício percorre difentes caminhos.

3.2.2. Algoritmo genético com busca local (GALS)

Os algoritmos genéticos ou *Genetic Algorithms* (GA) são algoritmos de busca e otimização baseados nos conceitos de genética e seleção natural [Pandey et al., 2014]. As definições do GA foram introduzidas por John H. Holland, em [Holland, 1992]. Os GA operam em uma população de soluções representadas por codificações e, durante a implementação do GA, para toda solução ou indivíduo é atribuído um valor de *fitness*, que serve para orientar as operações dos algoritmos genéticos [Pandey et al., 2014].

O princípio fundamental dos GA é que os indivíduos de uma determinada população disputam por recursos e, aqueles que obtiveram sucesso na disputa, acabam originando um maior número de descendentes, passando seus genes para as gerações seguintes. Ainda, os indivíduos das novas gerações podem ou não possuir melhores valores de *fitness* quando comparados a seus antecessores [Bajaj e Sangwan, 2019]. Em adição, cada um desses indivíduos representa uma solução referente ao problema estudado.

De acordo com [Yu e Gen, 2010], para GA há três características que podem ser consideradas fundamentais: é baseado em populações, que basicamente são um grupo de soluções e cada solução um indivíduo; *Fitness-oriented*, onde cada indivíduo tem uma representação genética e um valor de desempenho; e além das operações de cruzamento (*crossover*) permite mutações, uma vez que os indivíduos sofrerão variações para simular as mudanças sofridas por genes.

Aqui apresentamos um algoritmo híbrido de GA com Busca local (GALS) aplicado ao problema de minimização do *Makespan* cujo pseudo-código pode ser visto no Algoritmo 2.

O cromossomo é um vetor com sequência de tarefas alocadas em cada uma das máquinas, utilizou-se o método de chaves aleatórias [Bean, 1994], para formar a população inicial.

Algoritmo 2: Algoritmo GALS genérico

```
1 Entrada(arquivo instancias);
2 pop ← criaPopulacaoInicia();
3 pop ← BuscaLocal(pop);
4 repita
5   popPais ← selecionaIndividuoRecombina(pop)
6   popFilhos ← RecombinaMutacaoIndividuo(popPais)
7   popFilhos ← BuscaLocalTroca(popFilhos)
8   pop ← novaGeracao(pop, popFilhos)
9 até critério de parada satisfeito;
10 retorna melhorSolucao(pop)
```

Posteriormente as chaves são ordenadas, de forma crescente, definindo o ordenamento das tarefas que serão inseridas na solução, através de uma heurística gulosa, onde avalia-se em que máquina de qual fábrica a tarefa deve ser programada.

A seleção de indivíduos para gerar a nova população é feita pelo método de torneio para a escolha do primeiro cromossomo, pai 1, que dará origem a novos indivíduos, filhos. A seleção do segundo indivíduo, pai 2, é feita de forma aleatória. Eles serão recombinados em um ponto único gerando 2 filhos.

Na fase de mutação, que tem por finalidade aumentar a diversidade da população, novas chaves são sorteadas para 2% do número de genes. Após a mutação identifica-se qual o pior e o melhor indivíduo da nova geração, assim há a substituição do pior indivíduo da nova geração pelo melhor indivíduo da geração anterior, gerando assim a nova população.

Após a geração da nova população é feito uma Busca Local com troca, ou seja, a tarefa com maior tempo de processamento da máquina mais carregada é trocada por outra de outra máquina independente da fábrica. A partir disso a população inicial é totalmente atualizada.

Os parâmetros informados também são utilizados na aplicação do GALS ao problema de minimização do *Total Completion Time*, que tem a mesma estrutura do Algoritmo 2. Os parâmetros iniciais seguem os mesmos passos abordados no GLSA para minimização do *Makespan*, porém, para minimizar o *Total Completion Time* após o novo sorteio de chaves para genes é feito uma Busca Local com inserção, onde a cardinalidade das tarefas é considerada para que resultem em soluções factíveis, retirando tarefas da máquina mais carregada e inserindo nas demais máquinas de qualquer fábrica, levando em consideração a sua posição na ordem de processamento da tarefa.

A ordem em que a tarefa é inserida também é analisada pois, devido a forma que o tempo de finalização total é calculado, somando todos os tempos de processamento acumulados para cada tarefa em que uma máquina irá processar, influencia o valor total de processamento ao final.

O critério de parada estabelecido para ambos os casos analisados foi o número de gerações sem melhora no resultado sendo de no máximo 5 gerações e o número máximo de 20 gerações ao total.

Posterior à apresentação das heurísticas vai-se demonstrar e discutir os resultados obtidos no presente estudo.

4. Resultados computacionais

Os experimentos computacionais foram executados em Linguagem Julia utilizando um computador com Sistema Operacional Ubuntu 18.04, Intel Core i7, 1.8 Ghz com 8 GB de memória

RAM. O modelo matemático 1 foi processado em um computador com Sistema Operacional Linux Mint 19.2, processador Core i5, 2.5 Ghz e com 8 GB de memória RAM. Utilizou-se um tempo limite 1.800 segundos, para a execução dos Modelos 2 e 3 no CPLEX, para o Modelo 1 foi calculado um limitante inferior para as comparações.

4.1. Geração das instâncias

O conjunto das instâncias foram gerados de acordo com [Behnamian e Fatemi Ghomi, 2013]. As informações contidas em cada instância do conjunto são o número de tarefas representadas por n , o número de fábricas f , o número de máquinas em cada fábrica m_k , a velocidade das máquinas em cada fábrica v_k e o tempo de processamento para cada tarefa p_j .

Os algoritmos desenvolvidos foram divididos em 3 cenários e aplicados em 30 instâncias diferentes, baseados nos dados acima descritos. Cada cenário é composto por um total de 10 instâncias. Conforme a distribuição mostrada na Tabela 1. As instâncias divididas em Cenários permitem analisar os resultados de forma mais apropriada, tendo em vista as características dos mesmos.

Tabela 1: Configuração das instâncias

Dados de entrada	Cenário 1	Cenário 2	Cenário 3
F	3	3	3
n	50	50	100
v	{3,3,3}	{3,4,5}	{3,4,5}
p	instâncias 1 a 10	instâncias 11 a 20	instâncias 21 a 30

4.2. Lower Bound

Para comparar os resultados do GALS e da TS, foi calculado um *Lower Bound* para o *Makespan*, que auxilia na análise da qualidade das soluções heurísticas.

O *Lower Bound* (Limitante Inferior), é calculado de maneira independente do processamento, sendo considerado como um valor de referência, garantindo que seja menor ou igual ao *Makespan*. Utilizou-se o *Lower Bound* apresentado por [Behnamian e Fatemi Ghomi, 2013], obtido pela equação 18:

$$LB = \max \left(\frac{p_1}{v_1}, \frac{p_1 + p_2}{v_1 + v_2}, \dots, \frac{\sum_{i=1}^{M-1} p_i}{\sum_{h=1}^{M-1} v_h}, \frac{\sum_{i=1}^n p_i}{\sum_{h=1}^M v_h} \right) \quad (18)$$

O *Makespan* precisa ser pelo menos maior que o tempo necessário para que a máquina mais rápida execute a tarefa mais longa. Esse tempo representa o primeiro termo dentro do *max*. O restante dos primeiros termos $m-1$ é determinado da mesma forma. E para o último termo é um pouco diferente, pois, o tempo mínimo necessário para processar todas as n tarefas nas m máquinas, mantendo todas as m máquinas ocupadas exatamente a mesma quantidade de tempo.

Dessa forma, os modos de solução conseguem ser analisados quanto a sua eficiência em relação a qualidade da solução e ao custo computacional. Os resultados são apresentados e discutidos a seguir.

4.3. Resultados do modelo proposto para GALS e Busca Tabu

Os resultados mostrados nas Tabelas 2, 3 e 4 demonstram o comparativo dos valores encontrados com a execução da *Busca Tabu* (BT) e do GALS, ainda, acrescidos do *Lower Bound* (LB) e do tempos computacional do GALS, para o Modelo 1. Os tempos da BT não foram apresentados, pois ficaram abaixo de 0,015s em todas as instâncias. Também, é apresentado o GAP em relação ao LB da melhor solução entre BT e GALS.

Tabela 2: Resultados do Cenário 1

Inst.	LB	BT	GALS	Tempo(s)	Gap%
1	250,38	250,73	250,72	0,17	0,13
2	206,21	206,55	206,55	0,16	0,16
3	273,04	273,51	273,39	0,16	0,35
4	236,37	239,54	236,79	0,18	0,18
5	234,68	234,87	234,87	0,19	0,08
6	233,57	233,93	233,91	0,19	0,16
7	206,64	206,84	206,84	0,19	0,10
8	205,35	205,69	205,67	0,19	0,16
9	257,35	257,69	257,69	0,14	0,13
10	254,69	255,14	255,14	0,21	0,18
Méd.	235,83	236,45	236,16	0,18	0,14

Tabela 3: Resultados do Cenário 2

Inst.	LB	BT	GALS	Tempo(s)	Gap%
11	190,10	194,29	190,95	0,21	0,34
12	172,57	172,87	172,87	0,27	0,17
13	144,15	168,95	144,47	0,29	0,22
14	166,93	168,23	167,29	0,24	0,22
15	153,10	162,41	154,15	0,32	0,69
16	172,57	174,50	173,98	0,18	0,82
17	141,18	141,93	141,57	0,32	0,28
18	143,51	143,91	143,84	0,23	0,23
19	160,84	161,08	161,06	0,26	0,14
20	185,68	186,06	186,02	0,23	0,18
Média	163,06	167,43	163,62	0,25	0,34

Tabela 4: Resultados do Cenário 3

Inst.	LB	BT	GALS	Tempo(s)	Gap%
21	369,59	369,92	370,02	0,76	0,09
22	314,97	315,32	315,28	0,99	0,10
23	266,15	266,50	266,50	0,88	0,13
24	330,07	330,26	330,22	1,22	0,04
25	341,28	341,67	341,62	1,04	0,10
26	308,42	308,46	308,70	1,13	0,01
27	294,46	294,73	294,69	1,24	0,08
28	330,43	330,60	330,58	0,94	0,04
29	389,26	389,32	389,28	1,76	0,01
30	349,53	350,04	350,02	0,94	0,14
Média	329,42	329,69	329,69	1,06	0,08

Na Tabela 2, sendo o Cenário 1, no qual foram alocadas 50 tarefas em 3 máquinas com velocidades iguais em 3 fábricas, constatou-se que os resultados de GALS foram no mínimo iguais aos da *Busca Tabu*. Nesse ponto é importante destacar a instância número 4, que indica uma significativa melhora, em relação ao valor obtido pela BT, corroborada pelo valor de *Lower Bound* de referência.

No cenário 2, conforme consta na Tabela 3, foram alocadas 50 tarefas para 3, 4 e 5 máquinas com velocidades iguais. Ao observar as informações apresentadas em seguida, mostrou-se que as instâncias de número 14 e número 15 apresentaram consideráveis melhorias nos valores apontados pelo GALS. Seguido das demais instâncias com pequenas variações e a instância de número 2, que demonstrou o mesmo resultado da aplicação pela *Busca Tabu*.

A Tabela 4, por sua vez, apresentou o comparativo dos resultados obtidos para o Cenário 3, no qual foram alocadas 100 tarefas para 3, 4 e 5 máquinas com velocidades iguais. Em síntese,

os valores resultantes das últimas instâncias, indicou nas instâncias número 21 e número 26, que a Busca Tabu obteve melhores tempos de conclusão. Enquanto, nas demais instâncias, com exceção do número 2, que houve um empate, apontaram valores com alguma melhora com uso do GALS.

Estes resultados demonstram que as duas heurísticas se comportam muito bem, com Gaps, em relação ao LB, menores que 1% em todos os casos, sendo que a BT executa em um tempo ligeiramente menor que o GALS. Esses resultados indicam que as heurísticas encontraram valores muito próximos da solução ótima ou mesmo ela.

O *Total completion time* obtido a partir do algoritmo sugerido, realiza a soma acumulada, sendo assim, em sua execução testa vários movimentos de melhora, mas o executa apenas se houver algum ganho, uma única vez.

Tabela 5: Tabela 5 - Resultados do Cenário 1

Inst.	CPLEX Mod. 2	CPLEX Mod. 3	GALS	GAP%	Tempo(s)
1	6996,98	6996,98	6996,98	0,00	0,85
2	5766,68	5766,68	5766,68	0,00	0,98
3	7629,27	7629,27	7629,27	0,00	0,93
4	6602,19	6602,19	6602,19	0,00	1,04
5	6561,24	6561,24	6561,24	0,00	1,01
6	6535,80	6535,80	6535,80	0,00	1,07
7	5771,92	5771,92	5771,92	0,00	1,09
8	5734,84	5734,84	5734,84	0,00	1,10
9	7197,75	7197,75	7197,75	0,00	0,99
10	7118,18	7118,18	7118,18	0,00	0,97
Méd.	6591,48	6591,48	6591,48	0,00	1,00

Tabela 6: Tabela 6 - Resultados do Cenário 2

Inst.	CPLEX Mod. 2	CPLEX Mod. 3	GALS	GAP%	Tempo(s)
11	5619,14	5619,14	5619,14	0,00	0,85
12	5082,10	5082,10	5082,10	0,00	0,92
13	4254,19	4254,19	4254,19	0,00	0,98
14	4919,14	4919,14	4919,14	0,00	1,11
15	4523,36	4523,36	4523,36	0,00	1,00
16	5089,60	5089,67	5089,60	0,00	1,02
17	4181,78	4181,78	4181,78	0,00	0,99
18	4236,87	4237,20	4236,87	0,00	0,92
19	4751,98	4751,98	4751,98	0,00	0,92
20	5456,54	5456,54	5456,54	0,00	1,11
Méd.	4811,47	4811,51	4811,47	0,00	0,98

Tabela 7: Tabela 7 - Resultados do Cenário 3

Inst.	CPLEX Mod. 2	CPLEX Mod. 3	GALS	GAP%	Tempo(s)
21	19573,98	19575,00	19573,66	0,002	5,37
22	16671,31	16671,76	16670,73	0,003	5,78
23	14099,66	14100,34	14099,62	0,000	4,44
24	17466,61	17466,87	17466,41	0,001	5,90
25	18059,94	18061,41	18059,53	0,002	5,52
26	16327,65	16328,27	16327,00	0,004	4,61
27	15603,69	15605,51	15603,46	0,002	5,80
28	—	17492,90	17491,63	0,007	4,97
29	20620,46	20622,74	20620,43	0,000	6,64
30	18513,74	18515,15	18513,21	0,003	5,10
Méd.	17437,45	17443,99	17442,57	0,002	5,41

Nas Tabelas 5, 6 e 7 são apresentados os resultados para os três cenários com o objetivo de minimizar o *Total completion time*. Serão mostrados os resultados obtidos pelo *Solver* CPLEX, considerando os Modelos 2 e 3 apresentados, ressaltando que os valores obtidos não garantiram a solução ótima, porém todos eles tem um GAP do solver menor que 0,82% e como pode-se ver pelos Gaps percentuais em relação a heurística, quando são zero, indicam que encontrou-se a solução ótima. Um traço, mostra que o CPLEX não conseguiu resolver aquela instância.

Das Tabelas 5 e 6, pode-se ver que o GALS encontrou a solução ótima para todas as instâncias do Cenário 1 e 2, uma vez que o valor obtido pelo solver é igual ao da heurística.

Já pela Tabela 7, com os resultados do Cenário 3, percebe-se um Gap percentual, muito pequeno, sempre menor que 0,004%, mas com um tempo em torno de 5 vezes o gasto nos Cenários 1 e 2.

No Cenário 3, instância 28, observa-se que o CPLEX não conseguiu nenhum limitante para o problema dentro do limite imposto de 1800 segundos.

Pode-se notar então que as soluções do GALS para a minimização do TCT, se não são ótimas, estão muito próximas delas. Na sequência vai-se apresentar as considerações finais e alguns encaminhamentos de trabalhos futuros.

5. Considerações Finais

Este trabalho apresentou um Algoritmo Genético com busca local (GALS) e uma Busca Tabu (BT), para o problema de sequenciamento em máquinas paralelas idênticas em multi-fábricas com minimização do *Makespan* e ainda uma fórmula para cálculo de um limitante inferior para esse problema. Foi gerado um conjunto de 30 instâncias com diferentes configurações de máquinas por fábrica. Também adaptou-se o GALS para resolver o mesmo problema com o objetivo de minimizar o *Total completion time* (TCT) e apresentou-se um modelo matemático (Modelo 3) para o problema que permitiu que o CPLEX resolvesse todas as instâncias.

Os resultados demonstraram que tanto a BT quanto o GALS apresentaram muito bons resultados, tanto para a minimização do *Makespan* quanto do TCT, ficando sempre abaixo de 1% dos valores dos limitantes apresentados ou chegando na solução ótima quando se pode demonstrar isso. Tudo as custas de um pequeno esforço computacional.

Dentre as dificuldades encontradas no estudo, temos a falta dos dados do próprio problema, para que fosse possível uma comparação com a literatura. Finalmente, pode-se dizer que BT e GALS são eficientes e eficazes na solução do HMFPS para ambos os objetivos testados.

Como trabalhos futuros, pretende-se avaliar o problema para um parque de máquinas não relacionadas e com algumas relações de precedência, o que aproxima o mesmo da realidade encontrada no sistema produtivo. Também vai-se considerar a inclusão do problema de transporte entre as fábricas.

Referências

- Bajaj, A. e Sangwan, O. P. (2019). A systematic literature review of test case prioritization using genetic algorithms. *IEEE Access*, 7:126355–126375.
- Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing*, 6:154–160.
- Behnamian, J. e Fatemi Ghomi, S. M. T. (2013). The heterogeneous multi-factory production network scheduling with adaptive communication policy and parallel machine. *Information Sciences*, 219:181–196.
- Behnamian, J. e Fatemi Ghomi, S. M. T. (2016). A survey of multi-factory scheduling. *Journal of Intelligent Manufacturing*, p. 231–249.
- Chaouch, I., Driss, O. B., e Ghedira, K. (2020). A review of job shop scheduling problems in multi-factories. *International Journal of Operational Research*, 38(2):147–165.

- da Silva Xavier, A., Lins, S. L. S., e da Silva, A. A. (2020). Sequenciamento de tarefas em máquinas paralelas uniformes com técnicas de fluxo em rede: uma aplicação prática. *Brazilian Journal of Development*, 6(5):26354–26370.
- Fuchigami, H. Y. e Rangel, S. (2014). Uma análise de estudos de casos em sequenciamento da produção. *XLVI SBPO, Salvador-BA*, p. 159–170.
- Glover, F. (1989). Tabu search—part i. *ORSA Journal on Computing*, 1(3):190–206.
- Holland, J. H. (1992). Genetic algorithms. *Scientific American*, 267(1):66–73. ISSN 00368733, 19467087. URL <http://www.jstor.org/stable/24939139>.
- Huo, Y. (2019). Parallel machine makespan minimization subject to machine availability and total completion time constraints. *Journal of Scheduling*, 22(4):433–447.
- Joo, C. M. e Kim, B. S. (2017). Rule-based meta-heuristics for integrated scheduling of unrelated parallel machines, batches, and heterogeneous delivery trucks. *Applied Soft Computing*, 53(C): 457–476.
- Pandey, H. M., Chaudhary, A., e Mehrotra, D. (2014). A comparative review of approaches to prevent premature convergence in ga. *Applied Soft Computing*, 24:1047–1077. ISSN 1568-4946. URL <https://www.sciencedirect.com/science/article/pii/S1568494614003901>.
- Yazdani, M., Gohari, S., e Naderi, B. (2015). Multi-factory parallel machine problems: improved mathematical models and artificial bee colony algorithm. *Computers & Industrial Engineering*, 81:36–45.
- Yu, X. e Gen, M. (2010). *Introdução aos Algoritmos Evolutivos*. Springer Science.