UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO ACADÊMICO DO AGRESTE
NÚCLEO DE TECNOLOGIA
CURSO DE BACHARELADO EM ENGENHARIA DE PRODUÇÃO

JOSENILDO FERREIRA DA SILVA JÚNIOR

PROBLEMA DE ESCALONAMENTO DE MÁQUINAS PARALELAS COM EFEITOS DE DESGASTE E MANUTENÇÕES REPARADORAS: uma abordagem heurística

Caruaru
2022

JOSENILDO FERREIRA DA SILVA JÚNIOR

PROBLEMA DE ESCALONAMENTO DE MÁQUINAS PARALELAS COM EFEITOS DE DESGASTE E MANUTENÇÕES REPARADORAS: uma abordagem heurística

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Produção da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Barachel em Engenharia de Produção.

Área de concentração: Pesquisa Operacional.

Orientador: Prof. Dr. Walton Pereira Coutinho

Caruaru
2022

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Ferreira, Josenildo.
Problema de escalonamento de máquinas paralelas com efeitos de desgaste e manutenções reparadoras: uma abordagem heurística / Josenildo Ferreira. - Caruaru, 2022.
52

Orientador(a): Walton Coutinho
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro Acadêmico do Agreste, Engenharia de Produção, 2022.

1. Problemas de Programação da Produção. 2. Busca em Vizinhança Variável. 3. Problema de Otimização Multiobjetivo. I. Coutinho, Walton. (Orientação). II. Título.

620 CDD (22.ed.)

JOSENILDO FERREIRA DA SILVA JÚNIOR

PROBLEMA DE ESCALONAMENTO DE MÁQUINAS PARALELAS COM EFEITOS DE DESGASTE E MANUTENÇÕES REPARADORAS: uma abordagem heurística

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Produção da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Barachel em Engenharia de Produção.

Aprovada em: 24/10/2022

BANCA EXAMINADORA:

Prof. Dr. Walton Pereira Coutinho
(Orientador)
Universidade Federal de Pernambuco

Prof. Dr. Lúcio Câmara e Silva
(Examinador Interno)
Universidade Federal de Pernambuco

Prof. Dr. Luciano Carlos Azevedo da Costa
(Examinador Externo)
Universidade Federal da Paraíba

AGRADECIMENTOS

Fui muito bem orientado e aconselhado durante toda a minha graduação no Centro Acadêmico do Agreste por seu corpo docente, como os professores Osmar, Marcele e Thalles, então os agradeço por tudo que os senhores fazem por nós, seus alunos. Mas com certeza, dentre todos eles, eu sou eternamente grato aos professores Lúcio, Luciano e Walton.

Para atingirmos nossos objetivos, é necessário que nos dediquemos muito, mas sem pessoas para nos guiar durante esse processo, isso se torna extremamente difícil, beirando ao impossível.

Por exemplo, sem o professor Lúcio, eu não teria me dedicado à programação, algo fundamental para a área de Pesquisa Operacional (PO). Não teria avançado em aspectos vitais da PO, como meta-heurísticas, não teria me envolvido em casos teóricos e mesmo aplicações práticas bem legais, nem teria nenhuma publicação com meu nome.

Já sem o professor Luciano eu ainda estaria sofrendo muito para aprender melhores práticas na área de programação, demoraria muito para descobrir alguns atalhos para a modelagem de alguns problemas e sua implementação. Fora isso, eu estaria bem perdido sem o professor Luciano quando eu finalmente me deparasse com uma situação real que requisitasse uma solução a partir de uma ferramenta de PO.

Por fim, sem o professor Walton, para início de conversa eu não estaria estagiando na área de PO, não teria visto a importância de algumas práticas em programação, coisas bem básicas até, mas nunca tinha me dado conta. Também, na área de PO, foi muito importante para mim aquelas discussões sobre o modelo trabalhado, sobre formas de resolvê-lo, sobre prazos, sobre cobrir o necessário e não ser perfeccionista. Fora tudo isso, sem o professor Walton, sem apresentação de TCC esse período.

Sou muito grato também pelo auxílio financeiro dado durante a minha iniciação científica pela FACEPE.

Também, igualmente importante para a minha vida acadêmica são as pessoas que fazem parte da minha vida fora dela.

Mãe e pai, agradeço a eles por tornar tudo possível. Todas as minhas conquistas partem deles, todas as minhas dificuldades foram amenizadas por eles, e espero muito poder orgulhá-los. Fora isso, posso falar com tranquilidade que meus pais são meus melhores exemplos com relação a dedicação, seja nos estudos no caso de mãe, que tem o dom de surpreender a todos ao mostrar o quão longe ela chega, ou fora dele no caso de pai, com a garra e a persistência para montar o próprio negócio.

Deyse, junto de você eu pude crescer muito, amadurecer e me senti motivado a ser melhor. Lutas, dificuldades e momentos especiais, nós compartilhamos vários desses nos últimos anos, e hoje não sei como poderia ter vivido isso sem sua presença. Eu te amo Deyse, e espero que possamos ficar juntos até o fim das nossas vidas.

Izabel, não é para menos te colocar por aqui, você com certeza salvou minha vida, e agora,

espero muito que você consiga ter orgulho de mim. Saiba que todas as mudanças que surgiram na minha vida nos últimos 2 anos e daqui para frente, foram/serão possibilitadas de alguma forma por você.

Adriana, você é uma das minhas pessoas favoritas da vida, e eu não sei o que teria sido de mim sem você. Você participou de muitos dos momentos mais importantes da minha vida de 2017 para cá, seja ouvindo, dando conselhos, ou mesmo só me chamando de idiota. Por isso, eu te amo.

Agradeço também a Lucas, meu irmão, a Joseane, e todas as outras pessoas que, mesmo indiretamente, contribuíram também com este trabalho.

RESUMO

O escalonamento da produção possui um papel vital no funcionamento dos setores produtivos da sociedade. Para lidar com essa problemática, um dos possíveis caminhos é adotar modelos de Problemas de Programação da Produção (PPP), os quais se baseiam em programação matemática, meta-heurísticas e outras técnicas para encontrar uma boa programação conforme métricas próprias de desempenho. Este trabalho aborda o Problema de Escalonamento em Máquinas Paralelas (PEMP) não relacionadas, com o objetivo de minimizar o *makespan* e o de minimizar somatório dos atrasos e adiantamentos nas atividades de manutenção. Para resolver o problema, é proposta uma heurística multiobjetivo Busca em Vizinhança Variável Multiobjetivo (BVVMO) que se baseia em soma ponderada e utiliza a Busca em Vizinhança Variável para resolver o mesmo problema com diferentes pesos para os objetivos, além de uma heurística construtiva gulosa para gerar a solução inicial em uma região mais promissora de busca. O algoritmo BVVMO foi então comparado com os métodos NSGA-II e AUGMECON, já utilizados na literatura para resolver esse problema. A partir disso, verificou-se dentro das condições testadas que a BVVMO possui um desempenho superior ao NSGA-II, embora quando comparado com o AUGMECON, tenha uma performance consideravelmente inferior, mesmo para instâncias pequenas. Portanto, recomenda-se mais estudos para a resolução do problema em análise no trabalho.

Palavras-chave: Problemas de Programação da Produção; Busca em Vizinhança Variável; Problema de Otimização Multiobjetivo.

ABSTRACT

Scheduling plays a vital role in the functioning of the productive sectors of society. To approach this, a company can adopt Production Scheduling Problem models, which uses mathematical programming, meta-heuristics and other techniques to find a good programming according to their own performance metrics. This work addresses an unrelated Parallel Machine Scheduling Problem, with the objectives of minimize the makespan and the sum of earliness and tardiness in maintenance activities. To solve the problem, a Multiobjective Variable Neigborhood Search (MOVNS) heuristic is proposed that is based on weighted sum and uses Variable Neigborhood Search to solve the same problem with different weights to the objectives, in addition to a greedy constructive heuristic to generate the initial solution in a more promising search region. This research compares the MOVNS algorithm with the NSGA-II and AUGMECON methods, already used in literature to solve this problem. Under the tested conditions, this study concludes that MOVNS algorithm has a superior performance to NSGA-II. However, the comparison with AUGMECON shows a considerably lower performance, even for small instances. Therefore, further studies are recommended to solve the problem under analysis at work.

Keywords: Scheduling; Variable Neighborhood Search; Multiobjective Optimization Problem.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Segundo Corrêa, Gianesi e Caon (2019), um sistema de produção deve ser capaz de prometer os menores prazos para os clientes, e cumpri-los. Além disso, ele deve conseguir programar as atividades de produção, de modo a garantir que os recursos sejam utilizados nas atividades seguindo uma ordem de prioridade. Para atingir esses objetivos no chão de fábrica, ao mesmo passo que torna viável na produção decisões tomadas em níveis hierárquicos superiores, esses sistemas podem utilizar o *Kanban*, ou ainda modelos de Problemas de Programação da Produção (PPP) (DENNIS, 2008; PINEDO, 2016).

Os PPPs lidam com o processo de tomada de decisão de alocar os recursos às tarefas com base em uma ou mais métricas de performance. Sua modelagem depende dos objetivos pretendidos pelos gestores, do ambiente das máquinas e de características do processamento das tarefas e possíveis restrições práticas (LEUNG, 2004a).

O ambiente das máquinas é dependente da quantidade de máquinas, do roteiro de processamento das tarefas e da diferença na velocidade das máquinas. Dentre os tipos de ambientes, há ambientes com máquina única, onde todas as tarefas devem ser processadas em uma mesma máquina, e os de várias máquinas como, por exemplo, o *Job Shop Scheduling Problem*, no qual cada tarefa pode possuir um roteiro próprio de processamento nas máquinas e o *Flow Shop Scheduling Problem*, onde todas as tarefas possuem o mesmo roteiro de processamento (PINEDO, 2016).

Um outro ambiente de máquinas é o Problema de Escalonamento em Máquinas Paralelas (PEMP), o qual é o foco da presente pesquisa. No PEMP genérico, n tarefas devem ser finalizadas dentro do horizonte de planejamento, dependendo apenas de 1 processamento que pode ser realizado em qualquer uma das m máquinas disponíveis (CHENG; SIN, 1990).

Certas características são comumente assumidas em PEMPs, como: todas as máquinas disponíveis a todo momento; tempos de processamento constantes; e todas os dados da programação conhecidos desde seu início (CHENG; SIN, 1990; ZAROOK; ABEDI, 2014). Contudo, durante a modelagem do PPP, elas podem ser revistas para que se represente mais fielmente a realidade do processo. Como exemplo disso, há os processos de corte de madeira e extrusão, dado que as máquinas podem sofrer uma deterioração considerável à medida que realizam as tarefas, isto é, a velocidade de processamento das tarefas diminui com o uso da máquina. Por causa disso, as máquinas precisam passar por manutenções, seja com o intuito de recuperar a eficiência delas, ou mesmo para garantir o próprio funcionamento dos aparelhos. O problema surge porque a realização de certas atividades de manutenção tornam a máquina indisponível para processamento, possivelmente prejudicando os objetivos relativos a entrega das tarefas (ZAROOK; ABEDI, 2014).

Essa característica do problema é chamada na literatura como PEMP com atividades modificadoras de taxa de processamento (RMA, do inglês *Rate-Modifying Activities*), sendo possível

perceber que, dentro desse contexto, as decisões de produção e manutenção se tornam interdependentes. Logo, empresas que considerem essas atividades de forma relacionada podem obter vantagens competitivas (GEURTSEN et al., 2022).

Os PEMPs possuem objetivos geralmente baseados nos tempos de finalização das tarefas, como o *makespan* e o tempo total de finalização, baseados nos tempos de entrega das tarefas, como o máximo atraso e o somatório de atrasos (CHENG; SIN, 1990).

Há ainda cenários nos quais o PEMP possui mais de um objetivo a ser incluído na modelagem. Dentre os casos estudados na literatura: no setor de produção de garrafas de uma empresa, na qual se quer maximizar lucros, ao mesmo tempo que se quer minimizar a diferença na carga de trabalho entre duas máquinas (T'KINDT; BILLAUT; PROUST, 2001); na definição de esquemas de paradas de trem de uma cidade, que leva em consideração os custos da operação e a perda de tempo no tempo total de viagem dos passageiros (CHANG; YEH; SHEN, 2000), fora outras situações relatadas por T'kindt e Billaut (2006).

De forma semelhante, em um problema baseado em uma aplicação prática de uma empresa de fabricação de sacolas plásticas, o problema introduzido por Barreto et al. (2022) e analisado no presente trabalho lida com 2 objetivos conflitantes em um PEMP. A empresa na qual o estudo se baseou conta com algumas extrusoras para, além de outras funções, derreter uma mistura composta de plástico e outras matérias primas. Contudo, esse processo resulta no desgaste da máquina, que resulta na perda de eficiência ou até mesmo em falha do equipamento por submetê-lo a condições extremas de temperatura e pressão.

Logo, torna-se necessário a realização de manutenções periódicas nas extrusoras, que devem ser consideradas no planejamento da produção por ter um impacto direto na eficiência e funcionamento das máquinas, dado que possuem a capacidade de recuperar parte ou toda a capacidade da máquina. Com o propósito de representar a interdependência da produção e da manutenção, assim como para incluir no PPP os objetivos de ambos os setores, os autores modelaram o problema com os objetivos de minimizar o *makespan* e o maior atraso ou adiantamento das manutenções.

Para problemas multiobjetivo, passa a ser necessário adotar métodos próprios de resolução, como os métodos exatos do ε-Restrito e da soma ponderada ou as meta-heurísticas *Non-dominated Sorting Genetic Algorithm II* (NSGA-II) e *Two-Phase Local Search* (TPLS) (MARLER; ARORA, 2004; DEB et al., 2002; PAQUETE; STÜTZLE, 2003). Dado a presença de vários métodos de resolução para problemas, é importante investigar qual deles possui o melhor desempenho geral. Para isso, Barreto et al. (2022) utilizou o NSGA-II e o ε-Restrito a partir do AUGMECON. Contudo, o estudo de Barreto et al. (2022) possui limitações com relação aos experimentos, como a não comparação entre os métodos de resolução viáveis para instâncias maiores, como as meta-heurísticas, dado a complexidade do problema em análise.

Portanto, no presente trabalho busca-se comparar o desempenho da meta-heurística NSGA-II e do AUGMECON associado ao CPLEX Solver, com a Busca em Vizinhança Variável Multiobjetivo (BVVMO), proposta no presente estudo.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Desenvolver um algoritmo heurístico para a resolução eficiente de um PEMP com efeitos de desgaste e manutenções reparadoras.

1.1.2 Objetivos Específicos

Para alcançar o objetivo geral, é necessário atender os seguintes objetivos específicos:

- Realizar levantamento bibliográfico sobre otimização multiobjetivo, problemas de escalonamento e métodos de resolução exatos e heurísticos para problemas de otimização;
- Desenvolver uma heurística multiobjetivo baseada em Busca em Vizinhança Variável (BVV) para o problema estudado;
- Realizar levantamento de dados de instâncias na literatura para realização de experimentos;
- Comparar a meta-heurística desenvolvida com os algoritmos disponíveis na literatura para a resolução do problema em estudo, conforme métricas de problema multiobjetivo.

1.2 JUSTIFICATIVA

Conforme Leung (2004a), o uso de métodos para resolver PEMP pode reduzir custos nas empresas, o que ajuda a mantê-las competitivas frente a um mundo cada vez mais globalizado.

Já de acordo com Muluk, Akpolat e Xu (2003), o escalonamento da produção não só impacta o desempenho da empresa, como também é um dos mais importantes fatores dentro da gestão da cadeia de suprimentos no nível micro. Sendo assim, a utilização de métodos efetivos de escalonar tarefas podem trazer benefícios para toda a cadeia na qual a empresa está inserida.

Motivados por isso, Barreto et al. (2022) investigaram uma indústria do setor de produção de sacolas plásticas, sobre a qual a pesquisa se deteve ao processo de fabricação de bobinas a partir de material reciclado. Os autores então modelaram o PPP do processo como um PEMP não relacionadas, considerando RMA e a deterioração das máquinas, e os objetivos de minimizar o *makespan* e de minimizar os atrasos e adiantamentos das manutenções, algo não encontrado na literatura de PEMP com RMA (BARRETO et al., 2022). Para resolvê-lo, foram utilizados o AUGMECON e a meta-heurística NSGA-II.

Contudo, não foi possível verificar o desempenho do NSGA-II para instâncias maiores dado a complexidade do problema para resolvê-lo com o auxílio de métodos exatos no AUGMECON. Além disso, segundo Geurtsen et al. (2022), apesar que dentre os PPPs que consideram as RMA, PEMP com RMA ser a área mais estudada do ramo, o uso de soluções criadas a partir

de meta-heurísticas é recente, e não há estudos que realizem a comparação entre diferentes os métodos de resolução tratados nos demais estudos.

Logo, ao propor o uso de da meta-heurística BVVMO e comparar seu desempenho com os demais métodos da literatura, o presente trabalho pretende preencher um *gap*, expandindo o campo de PEMP ao buscar qual o melhor método dentre os selecionados para resolver o problema de Barreto et al. (2022). Desse modo, busca-se facilitar a obtenção de benefícios para a cadeia de suprimentos a qual organizações que possuem processos semelhantes ao utilizado como base estão inseridas.

Quando considerado apenas o objetivo do *makespan*, o problema em análise pode ser visto como uma generalização do problema de duas máquinas paralelas idênticas com o objetivo de minimizar o *makespan*. Dado que tal problema é considerado $\mathcal{NP}$-difícil (BRUCKER, 2007), o problema em análise se torna de maior complexidade computacional ao incluir tempos de processamento não constantes em um contexto de otimização multiobjetivo. Dessa forma, justifica-se a utilização de métodos heurísticos, assim como a busca por melhores heurísticas para resolvê-lo.

2 REFERENCIAL TEÓRICO

2.1 PROGRAMAÇÃO INTEIRA MISTA

Conforme Morabito (2008), a Pesquisa Operacional utiliza métodos científicos para auxiliar na tomada de decisão, tendo aplicação em diversas áreas, como na logística e na produção, em situações onde os recursos são escassos. Dentre suas áreas, há análise de decisão, teoria dos jogos, teoria das filas e programação matemática (HILLIER; LIEBERMAN, 2006; TAHA, 2008).

A Programação Matemática (PM) utiliza modelos matemáticos para representar uma versão tratável de problemas da realidade. Para isso, o modelo conta com variáveis de decisão, que representam decisões quantificáveis relacionadas ao problema, sendo elas postas em relações matemáticas, como o objetivo do modelo a ser maximizado ou minimizado. Os modelos são dependentes de parâmetros para representar uma instância de um problema, e ao ser resolvida, gera-se uma ou mais soluções para a instância. Dentre os tipos de problemas abordados na literatura, há a os problemas de programação linear e os de programação inteira. (ARENALES et al., 2011; HILLIER; LIEBERMAN, 2006).

A Programação Linear (PL) trata de problemas de PM nos quais todas as equações do modelo são lineares e as variáveis podem assumir valores fracionários. Para resolver problemas de PL, há métodos já consagrados na literatura, que possuem a capacidade de lidar até com problemas de grande porte, como o simplex, um algoritmo iterativo que analisa as soluções viáveis do problema de forma eficiente, sendo um método exato, isto é, desconsiderando condições de parada impostas pelo usuário, ele finaliza apenas ao encontrar a solução ótima do problema. (HILLIER; LIEBERMAN, 2006; TAHA, 2008).

Já a PI se diferencia da PL por restringir todas - PI pura - ou parte - PI mista - das variáveis a serem inteiras. É importante mencionar que, embora a PI tratada no presente trabalho possa ser chamada de programação linear inteira, foi adotado essa nomenclatura de Hillier e Lieberman (2006). Por causa da restrição de integralidade das variáveis, alguns problemas de PI se tornam extremamente complexos e passa a ser necessário utilizar estratégias e outros métodos de resolução, não sendo possível resolvê-los utilizando o simplex por exemplo (GOLDBARG; GOLDBARG; LUNA, 2016; HILLIER; LIEBERMAN, 2006).

Apesar disso, dentre as formas de resolução de problemas de PI, é possível adaptar métodos de PL ao recorrer a algumas noções, como as de separação e de relaxação. Na separação, um problema é subdividido em problemas menores sob certas condições. Já na relaxação, algumas restrições do problema são eliminadas, como as de integralidade, e aos poucos durante a resolução elas são adicionadas, de modo a finalizar com uma solução que respeite todas as restrições. (GOLDBARG; GOLDBARG; LUNA, 2016).

Uma das adaptações do simplex utilizando, dentre outras noções, a relaxação e a separação

para a resolução de problemas de PI é o *Branch-and-Bound* (BB) (GOLDBARG; GOLDBARG; LUNA, 2016). O BB inicia com a resolução de um problema com a integralidade relaxada, sendo então ele ramificado em vários problemas que restringem as variáveis inteiras com relações de desigualdade. Sendo assim, o BB resolve um problema de PI a partir da busca em sucessivos problemas de PL. De forma semelhante, o algoritmo de plano de corte inicia com a solução ótima do problema de PI relaxado em um de PL, sendo então adicionadas restrições de corte para limitar a região de soluções viáveis. Por fim, o *branch-and-cut* torna o BB mais eficiente a partir de sua hibridização como algoritmo plano de corte. (ARENALES et al., 2011; TAHA, 2008).

Apesar da vantagem ao se encontrar uma solução ótima para o problema, nem sempre sua utilização é possível devido ao tempo de resolução elevado dos métodos exatos para problemas complexos ou de grande porte. Sendo assim, nesses casos é necessário recorrer a algoritmos aproximativos ou heurísticos e geralmente específicos para um tipo de problema, que não visam encontrar a solução ótima, mas sim uma boa solução em tempo hábil para a situação, os quais são abordados na seção a seguir (HILLIER; LIEBERMAN, 2006).

2.1.1 Métodos heurísticos de resolução

De acordo com Goldbarg, Goldbarg e Luna (2016), heurística é uma técnica aproximativa de busca de boas soluções no espaço de solução de problemas de otimização. Apesar de não possuir a capacidade de garantir que a solução encontrada seja ótima, as heurísticas possuem a capacidade de encontrar soluções em um tempo razoável, sendo comumente utilizadas para resolver problemas $\mathcal{NP}$-difíceis. $\mathcal{NP}$-difícil trata-se de uma classe de problemas que podem não possuir um algoritmo que os resolva em tempo polinomial, tornando-os possivelmente intratáveis (LEUNG, 2004b).

No início, eram criadas estratégias diferentes para resolução através de heurísticas de um tipo de problema, sendo difícil sua adaptação para demais problemas. Agora, as heurísticas podem ser construídas a partir de meta-heurísticas, estas que correspondem a uma arquitetura geral de regras que não são limitadas a um tipo de problema apenas, tornando mais fácil a criação de heurísticas. (GOLDBARG; GOLDBARG; LUNA, 2016; ÓLAFSSON, 2006).

Conforme Ólafsson (2006), meta-heurísticas são destinadas a resolver problemas que outros métodos não obtiveram uma resposta satisfatória, seja por ineficiência ou por ineficácia. Para isso, esses métodos buscam uma boa relação entre esforços de diversificação e de intensificação da região de soluções. Diversificação trata da capacidade da meta-heurística em encontrar regiões de busca promissoras, enquanto que os esforços de intensificação realizam a busca pelas melhores soluções em uma área promissora. (TALBI, 2009).

As meta-heurísticas podem ser classificadas a partir do tipo de busca realizada: a partir de uma população de soluções que vai evoluindo durante o processo de busca, a qual possui um foco maior na diversificação, como os Algoritmo Genético (AG) e colônia de abelhas, ou a partir de estruturas de vizinhança a ser alterarem uma solução, que possui um foco maior na

intensificação, como a busca tabu, o recozimento simulado, a Busca em Vizinhança Variável e a busca local iterada (TALBI, 2009).

Os métodos de busca baseados em estruturas de vizinhança podem ser representados a partir do algoritmo 1.

Algoritmo 1: Meta-heurísticas baseadas em estrutura de vizinhança

Entrada: Solução inicial s_0

```
1 t = 0;
2 enquanto O critério de parada não for satisfeito faça
3 |   Gerar(N(s_t));
4 |   s_{t+1} ← Selecionar(N(s_t));
5 |   t = t + 1;
6 fim
```

Saída: Melhor solução encontrada.

A partir de uma solução inicial s_0, inicia-se uma busca iterativa no espaço de soluções utilizando o conceito de vizinhança $N(s)$, bem como uma regra de seleção de solução a partir de um conjunto, estabelecido pelo usuário, enquanto o critério de parada definido não for satisfeito. Uma vizinhança $N(s)$ de uma solução s pode ser definida como um conjunto de soluções obtidas a partir de uma operação aplicada a s. Dessa forma, a melhor solução encontrada durante a busca é retornada. (GOLDBARG; GOLDBARG; LUNA, 2016; TALBI, 2009).

Em contrapartida, ao invés de uma busca migrando de foco de solução em solução, métodos populacionais realizam migrações entre conjuntos de soluções. As meta-heurísticas populacionais podem ser representadas pelo algoritmo 2.

Algoritmo 2: Meta-heurísticas baseadas em populações

```
1 Gerar população inicial P_0;
2 t = 0;
3 enquanto O critério de parada não for satisfeito faça
4 |   Gerar(P'_t);
5 |   P_t ← Selecionar população(P_t ∪ P'_t);
6 |   t = t + 1;
7 fim
```

Saída: Melhor solução encontrada.

As meta-heurísticas populacionais começam gerando uma população inicial, sendo seguida por um procedimento iterativo, no qual uma nova população P'_t será gerada, comparada com a população da iteração P_t, e então, os indivíduos de ambas as populações serão selecionados para compor a próxima população P_{t+1}. (TALBI, 2009).

O AG pode ser definido como uma meta-heurística populacional que mimetiza a evolução natural das espécies para a busca de soluções. Sendo assim, seu funcionamento ocorre como ilustrado no algoritmo 2, com a especificidade que, para a geração da população P'_t, são utilizados os operadores de seleção de indivíduos mais aptos, com base na função *fitness*, relacionada

à função objetivo, de cruzamento desses indivíduos para a geração da nova população, e de mutação, que pode modificar indivíduos da população gerada por cruzamento. (WHITLEY, 2019).

Problemas de otimização geralmente possuem apenas um único objetivo, sendo então a meta de métodos de resolução encontrar uma solução com o melhor ou um bom valor para essa métrica adotada. Em contrapartida, quando há 2 ou mais objetivo conflitantes em um problema de PM, a Otimização Multiobjetivo (OM) busca várias soluções, dado que não haverá uma única solução que seja melhor que as demais em todos os objetivos (MAVROTAS, 2009). Sendo assim, a próxima seção se estenderá na discussão sobre OM, tratando de conceitos necessários para a área, assim como adaptações em métodos exatos e em meta-heurísticas para tornar adequadas para resolver tais problemas.

2.2 OTIMIZAÇÃO MULTIOBJETIVO

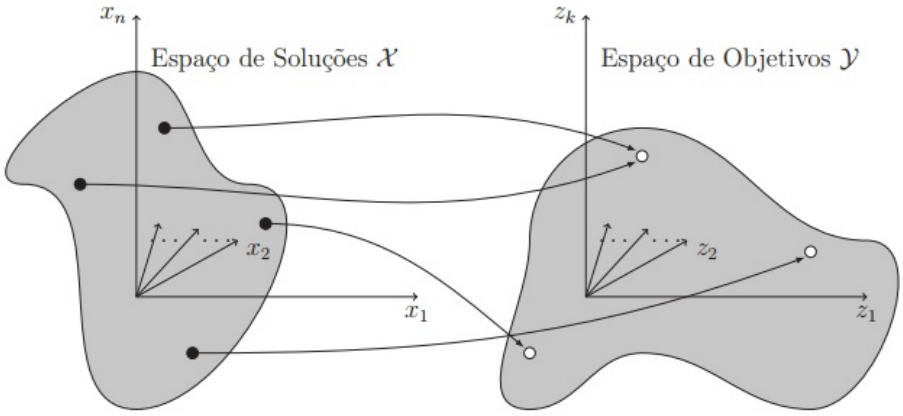
O Problema de Otimização Multiobjetivo (POM) pode ser definido com a seguinte formulação.

$$\min \mathbf{z}(\mathbf{x}) = [z_1(\mathbf{x}), z_2(\mathbf{x}), \ldots, z_k(\mathbf{x})]^T \tag{2.1}$$

$$x \in \mathcal{X} \tag{2.2}$$

Onde $\mathbf{z}$ é um vetor de k objetivos a serem minimizados, $\mathbf{x}$ corresponde a um vetor de n variáveis de decisão, limitadas ao espaço viável $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^n | g(\mathbf{x}) \leq 0, h(\mathbf{x}) = 0\}$. É possível ainda definir o conjunto $\mathcal{Y} = \{\mathbf{z}(\mathbf{x}) \in \mathbb{R}^p | \mathbf{x} \in \mathcal{X}\}$ como a imagem de $\mathcal{X}$ no Espaço dos objetivos. Desse modo, o modelo (2.1)-(2.2) pode ser representado conforme a Figura 1.

Figura 1 – Relação entre o Espaço de Soluções e o Espaço de Objetivos



Fonte: COSTA (2015).

Dado a característica perceptível na Figura 1 de que diferentes soluções, representadas por pontos em $\mathcal{X}$, podem possuir desempenhos diferentes para cada objetivo em $\mathcal{Y}$, passa a ser

necessário definir para POM conceitos para realizar a comparação entre as soluções, discutidos por Marler e Arora (2004, p. 370-372), Ehrgott (2005, p. 24, 34) e Talbi (2009, p. 311).

Definição 1 *Ótimo de Pareto ou eficiente e não eficiente: Uma solução* $\boldsymbol{x^* \in \mathcal{X}}$ *é um ótimo de Pareto ou eficiente se e somente se não existir outra solução* $\boldsymbol{x \in \mathcal{X}}$ *tal que* $\boldsymbol{z(x) \leq z(x^*)}$ *e* $z_t(\boldsymbol{x}) < z_t(\boldsymbol{x^*})$ *para ao menos uma função. Caso contrário,* $\boldsymbol{x^*}$ *é não eficiente.*

Desse modo, uma solução é ótima de Pareto conforme definição 1 se, a partir de mudanças nos valores das variáveis de decisão, não dá para melhorar um objetivo sem piorar seu desempenho nos demais. Contudo, nem sempre os métodos dedicados a POM consegue obter soluções classificadas como ótimas de Pareto, sendo assim:

Definição 2 *Ótimo de Pareto fraco: Uma solução,* $\boldsymbol{x^* \in \mathcal{X}}$*, é um ótimo de Pareto fraco se e somente se não existir outra solução* $\boldsymbol{x \in \mathcal{X}}$ *tal que* $\boldsymbol{z(x) < z(x^*)}$ *em todas as funções.*

Em outras palavras, uma solução $x^* *$ é definida como ótima de Pareto fraca quando não há uma solução x que possua desempenho superior em todos os k objetivos. É válido ressaltar que todos os ótimos de Pareto são ótimos de Pareto fracos, embora o oposto não seja sempre verdadeiro.

Definição 3 *Conjunto de soluções eficientes: o conjunto de pontos eficientes é chamado de conjunto de soluções eficiente, denotado por* $\mathcal{X}_E$*.*

Definição 4 *Pontos não dominados e dominados. Um vetor de funções objetivo,* $\boldsymbol{z(x^*) \in \mathcal{Y}}$ *é não dominado se e somente se não existir outro vetor* $\boldsymbol{z(x) \in \mathcal{Y}}$ *tal que* $\boldsymbol{z(x) \leq z(x^*)}$ *com pelo menos um* $z_t(\boldsymbol{x}) < F_t(\boldsymbol{x^*})$*. Caso Contrário,* $\boldsymbol{z(x^*)}$ *é dominado.*

Sendo assim, a seguinte definição trata da imagem das soluções eficientes.

Definição 5 *Fronteira de Pareto: a imagem do conjunto de soluções eficientes pode ser definida como fronteira de Pareto.*

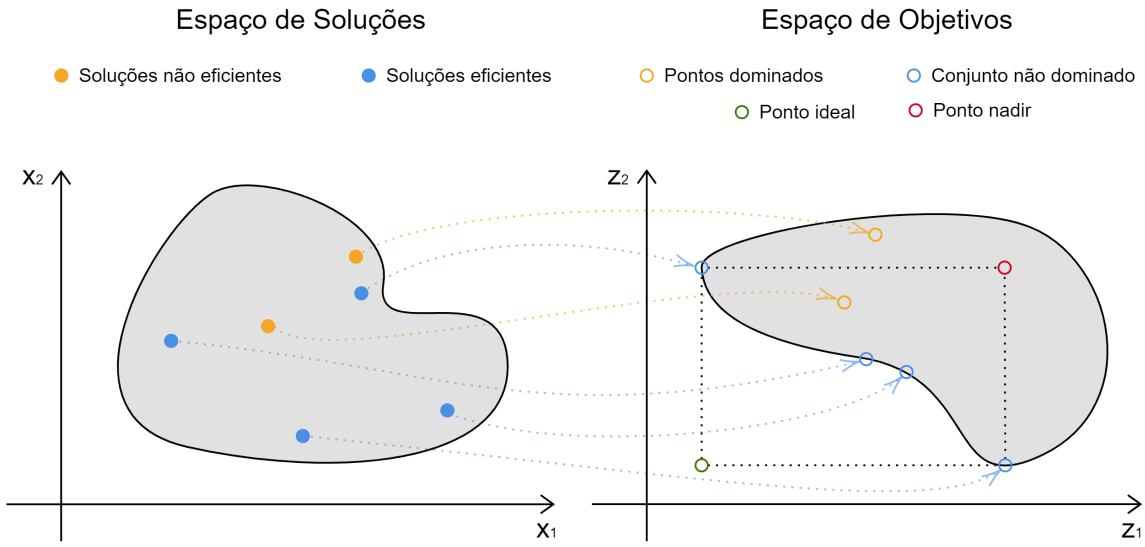
Por fim, métodos para resolver POM podem se utilizar dos conceitos de limites inferior e superior, bem como ponto Ideal e ponto Nadir, para delimitar os intervalos possíveis na fronteira de Pareto.

Definição 6 *Ponto ideal: o ponto* $y^I = (y_1^I, \ldots, y_k^I)$*, dado por* $y_i^I = \min_{\boldsymbol{x} \in \mathcal{X}} z_k(\boldsymbol{x})$*, é chamado de ponto ideal.*

Definição 7 *Ponto nadir: o ponto* $y^N = (y_1^N, \ldots, y_k^N)$*, dado por* $y_i^N = \max_{\boldsymbol{x} \in \mathcal{X}_E} z_k(\boldsymbol{x})$*, é chamado de ponto nadir.*

Para um problema com os objetivos z_1 e z_2 e as variáveis de decisão x_1 e x_2, as definições acima podem ser ilustradas a partir da Figura 2.

Figura 2 – Definições Multiobjetivo



De acordo com Miettinen (2003), é vasta a utilização da figura do decisor para a resolução de um POM, o qual terá suas relações de preferência incluídas no modelo. Já o momento que isso ocorre pode ser diferente para o tipo de método adotado para sua resolução:

- O decisor pode ter suas preferências explícitas no início da modelagem com o uso de métodos *à priori*;
- Já se isso só ocorre após a geração das soluções do problema, o método é classificado como *à posteriori*. O foco desse tipo de método é na construção do conjunto de soluções eficientes, não oferecendo formas de incluir as preferências do decisor na modelagem (ALMEIDA et al., 2015);
- Por fim, caso o decisor for agregando ao modelo suas preferências durante o processo de resolução, ele é chamado de *iterativo*.

Assim como no estudo realizado por Barreto et al. (2022), o foco deste trabalho é em métodos *à posteriori*. É importante ressaltar ainda que dado que o foco desses métodos é encontrar um conjunto de soluções eficientes do problema, a qualidade do método é medido por métricas próprias, como a métrica da qualidade, a quantidade de soluções não dominadas, dentre outros (AFZALIRAD; REZAEIAN, 2017).

A próxima seção aborda alguns dos métodos para a resolução de POM.

2.3 MÉTODOS DE RESOLUÇÃO PARA PROBLEMAS DE OTIMIZAÇÃO MULTIOBJETIVO

O método da soma ponderada é um dos métodos mais utilizados na literatura de POM. A partir dele, busca-se otimizar o problema mono-objetivo abaixo. (MARLER; ARORA, 2004).

$$\min \sum_{i=1}^{k} w_i \, f_i \tag{2.3}$$

$$x \in \mathcal{X} \tag{2.4}$$

Onde w_i representa o peso da função objetivo i, e f_i, a própria função objetivo, comumente normalizada.

Provado por Ehrgott (2005), a solução do problema (2.3)-(2.4) gerará sempre soluções ótimas de Pareto fracas quando todos os pesos w_o forem não negativos. Desse modo, para a geração do conjunto de soluções fracamente eficientes, o problema será resolvido várias vezes com diferentes combinações de pesos.

Dentre as desvantagens relatadas do método da soma ponderada destaca-se que ele não é capaz de gerar soluções para áreas não convexas da fronteira de Pareto, além de serem relatadas dificuldades com a definição dos pesos. (MARLER; ARORA, 2004; EHRGOTT, 2005).

Já o método do ε-Restrito busca resolver o POM selecionando um objetivo para minimizar, enquanto os demais são tratados como restrições. O ε-Restrito segue formulado abaixo. (MAVROTAS, 2009).

$$\min z_1 \tag{2.5}$$

$$z_i \geq \varepsilon_i \qquad \forall i \in \{2, 3, \ldots, k\} \tag{2.6}$$

$$x \in \mathcal{X} \tag{2.7}$$

Onde o intervalo $\{2, 3, \ldots, k\}$ corresponde ao conjunto de objetivos tratados como restrições, e e_i, ao limite inferior para cada função objetivo $i \in \{2, 3, \ldots, k\}$. O intervalo assumido por cada e_i será definido com o auxílio dos pontos nadir e ideal, e a partir de diferentes combinações de limites inferiores é criado o conjunto de soluções eficientes.

Assim como o método de soma ponderada, soluções do problema (2.5)-(2.7) garantem uma solução fracamente eficiente, embora seu uso possua algumas vantagens com relação ao último método, como não ser necessário que a fronteira de Pareto do problema seja convexo. Ainda há versões melhoradas do ε-Restrito, como o AUGMECON, o qual lida com algumas das desvantagens do método, como a dificuldade na definição do intervalo para cada objetivo, além de tornar todas as soluções encontradas pelo AUGMECON eficientes. (EHRGOTT, 2005; MAVROTAS, 2009).

Dessa forma, nota-se que a resolução pelos métodos descritos na presente seção envolvem a resolução do mesmo problema várias vezes com pequenas diferenças, seja no peso das funções objetivos em (2.3), ou ainda nos limites inferiores em (2.6), para desse modo encontrar o conjunto de soluções eficientes.

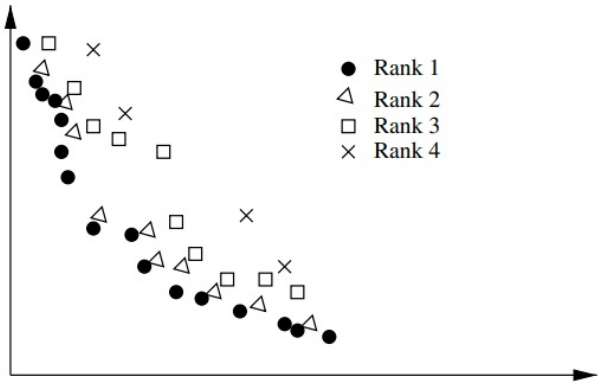
Contudo, como já descrito na seção anterior, se alguns problemas com apenas 1 objetivo necessitam recorrer a métodos heurísticos para encontrar uma boa solução em um tempo viável, recorrer a tais métodos se torna ainda mais justificável no contexto de POM. Dentre as metaheurísticas utilizadas para POM, há o *Non-dominated Sorting Genetic Algorithm-II* e o *Two-Phase Local Search*, os quais serão detalhados a seguir.

2.3.1 *Non-dominated Sorting Genetic Algorithm II* (NSGA-II)

Introduzido por Deb et al. (2002), o NSGA-II é uma adaptação do AG para POM, que consiste na comparação entre soluções utilizando conceitos de OM através dos procedimentos *non-dominated sorting* e *crowding distance*.

O *non-dominated sorting* busca agrupar todas ou parte das soluções de uma população nos seus respectivos frontes, como ilustrados na Figura 3.

Figura 3 – Frontes do NSGA-II



Fonte: adaptado de TALBI (2009).

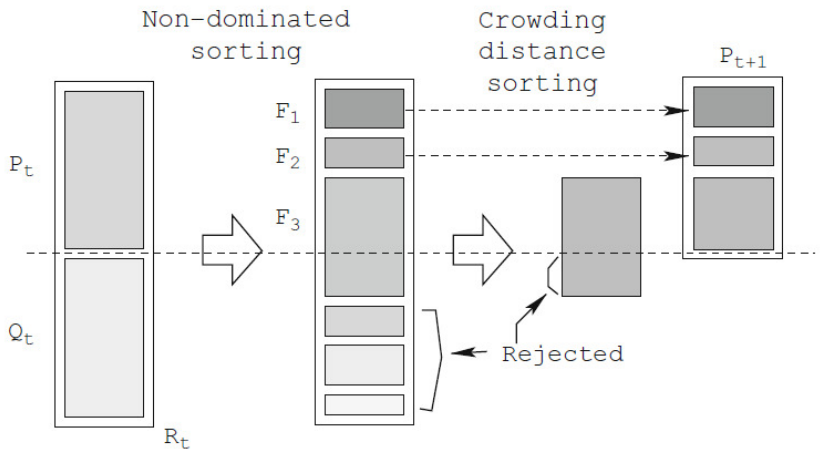
Um fronte F_i corresponde ao conjunto de soluções que são eficientes com relação a F_i e a frontes posteriores F_j, $\forall i < j$, e são não eficientes com relação a frontes anteriores F_k, $\forall i > k$. Por sua vez, o ranking r_l de uma solução l representa o fronte que a solução se encontra. Para isso, é realizada uma comparação par a par entre as soluções verificando relações de dominância.

Já *crowding distance* é utilizado no NSGA-II para comparar soluções de um mesmo fronte. Esse procedimento calcula a densidade das soluções nas proximidades de uma solução utilizando o perímetro de um cuboide, com o intuito de estimular a escolha de regiões menos densas, sendo um esforço pela diversificação.

Dessa forma, o processo de seleção recorre a ambos os procedimentos para comparar as soluções sob uma ótica multiobjetivo. A partir disso, o NSGA-II segue parecido com o AG,

gera-se uma população inicial P_0 de tamanho N, a qual será submetida em um processo iterativo aos operadores de seleção por torneio binário, de cruzamento e de mutação, ambas estabelecidos pelo usuário, para gerar uma nova população Q_t de tamanho N. Quanto a etapa de atualização da população P_{t+1}, ela pode ser ilustrada pela Figura 4.

Figura 4 – Procedimento do NSGA-II



Fonte: DEB (2011).

É definida a população $R_t = P_t \cup Q_t$, de tamanho $2N$, a qual será submetida ao *non-dominated sorting* para selecionar as melhores soluções conforme seus rankings para entrar na população P_{t+1}. A identificação dos frontes para tal ocorre até que a quantidade de soluções até último fronte investigado F_o for menor que N. Quando o último fronte F_{o+1} identificado, exemplificado na Figura 4 por F_3, possui mais soluções que a quantidade de vagas ainda disponível em P_{t+1}, é necessário recorrer ao *crowding distance sorting* para selecionar quais de suas soluções entrarão para a população da próxima iteração P_{t+1}.

2.3.2 *Two-Phase Local Search* (TPLS)

Proposto por Paquete e Stützle (2003) para problemas bi-objetivo, o TPLS se utiliza se 2 fases para encontrar um conjunto próximo à fronteira de Pareto. Conforme os autores, o método busca ser de fácil compreensão, ao mesmo tempo que robusto, flexível e eficiente (PAQUETE; STÜTZLE, 2003). Diferente do NSGA-II, trata-se de uma meta-heurística com busca a partir de uma única solução.

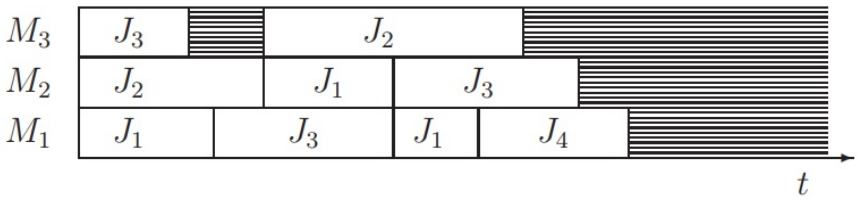
A primeira fase consiste em encontrar uma boa solução com base em apenas um dos objetivos, que pode ser realizada através de uma busca local a partir de uma solução aleatória, ou ainda outros métodos de otimização para problemas mono-objetivo, com uma boa performance para o problema em questão. A partir da solução encontrada, a segunda fase aplicará procedimentos de busca local, tendo o critério de seleção da solução da vizinhança como base uma soma ponderada dos objetivos, expressa em (2.3). Para encontrar as diferentes soluções do conjunto, a função (2.3) terá os seus pesos modificados durante as iterações do método.

2.4 PROBLEMAS DE ESCALONAMENTO

Responsáveis por orientar decisões de curtíssimo prazo, problemas de escalonamento são problemas de gerenciamento de menor hierarquia dentro de empresas (CORRêA; GIANESI; CAON, 2019). Além de auxiliar setores de produção, sua aplicação ocorre comumente para auxiliar a logística, empresas de serviço, dentre outros (PINEDO, 2016).

Problemas de escalonamento definem a ordem de processamento de n tarefas em m máquinas, respeitando o roteiro de cada tarefa J_i pelas m máquinas e demais restrições, de modo a atingir bons resultados para a organização com base em um ou mais critérios (LEUNG, 2004a). Uma solução do problema de escalonamento pode ser representada por gráficos de Gantt, como ilustrado pela Figura 5, que mostra no tempo a sequência de tarefas a ser executada por cada máquina.

Figura 5 – Gráfico de Gantt orientado por máquinas



Fonte: BRUCKER (2007).

Essa classe de problemas pode ser retratada pelo formato $\alpha|\beta|\gamma$, conforme abordado por Pinedo (2016, p. 14-18) e Geurtsen et al. (2022, p. 2-4).

O tipo padrão do problema de escalonamento é definido por α, sendo dependente da quantidade de máquinas, da velocidade delas, e do roteiro que as tarefas fazem para serem finalizadas. Dentre os tipos, é possível listar:

- *Single Machine Scheduling Problem* (1) Cenário no qual as n tarefas são processadas em 1 máquina.
- Problema de escalonamento em máquinas paralelas idênticas (Pm) As n tarefas requerem um único processamento, que pode ser realizada em qualquer uma ou em um subconjunto das m máquinas. Neste tipo, todas as máquinas capacitadas de processar uma tarefa i realizam a operação no mesmo tempo p_i.
- Problema de escalonamento em máquinas paralelas uniformes (Qm) Trata-se de uma generalização de Pm, na qual cada máquina k dentre as m máquinas possuem uma velocidade de processamento v_k, sendo o tempo de realização da tarefa j na máquina k igual a $p_{jk} = p_i\, v_k$.
- Problema de escalonamento em máquinas paralelas não relacionadas (Rm) Generaliza Qm quando estabelece que não há relação entre o tempo p_jk de processamento da tarefa j na máquina k e um parâmetro de velocidade v_k.

- *Flow Shop Scheduling Problem* (Fm) Generalização do problema de máquinas paralelas, na qual todas as n tarefas precisa passar por processamento em todas as m máquinas seguindo um mesmo roteiro de processamento.

- *Job Shop Scheduling Problem* (Jm) Generaliza o Fm ao permitir que cada tarefa j dentre as n tarefas siga um roteiro próprio de processamento pelas m máquinas.

Já a restrições e características das tarefas são entradas de β. Dentre possíveis especificidades do problema:

- Preempção ($prmp$) As tarefas podem ser interrompidas durante seu processamento para serem resumidas posteriormente.

- Produção em lote ($batch$) Cada máquina pode processar as tarefas que compõe um lote simultaneamente, sendo o tempo de finalizar um lote igual ao tempo de finalizar a tarefa mais longa que a ele pertence.

- Tempos de setup dependentes da sequência (s_{ijk}) Quando os tempos de setup são dependentes da ordem das tarefas i e j, ou ainda quando são dependentes da máquina k.

- Atividades modificadoras de taxa (RMA, do inglês *Rate-Modifying Activities*) Quando os tempos de realizar alguma atividade varia conforme a realização de outras atividades, sendo geralmente relacionadas a manutenções ou a tarefas. Dentre os tipos de RMA, há:

  - RMA_{j-op-o} Cada tarefa j possui seu tempo de processamento desgastado dependente de alguma op relacionada ao momento que ela é realizada no escalonamento, podendo esse desgaste ser revertido o vezes, que representa o número de manutenções. Pode ser declarado em op os tipos de desgaste: tempo - $time$ -, posição - pos - ou potência - pot.
  - RMA_{m-op} Funciona de forma semelhante a tarefas, contudo o desgaste é relacionado ao tempo de se realizar uma manutenção m.

- Data devida de tarefa (d_j) Cada tarefa j possui uma data a ser finalizada. As datas devidas podem ser iguais para todas as tarefas, sendo então declarado $d_j = d$. Já caso haja data devida para cada manutenção m no contexto de RMA, declaro d_m.

Por fim, os objetivos do escalonamento são expressos em γ, dentre os quais

- *Makespan* (C_{max}) O tempo no qual a última tarefa da programação em todas as máquinas foi finalizada.

- Tempo total de conclusão ($\sum C_j$) A soma do tempo total de finalização das tarefas. Para versão ponderada do objetivo $\sum w_j C_j$.

- Somatório de Atrasos e adiantamentos das tarefas ($\sum(E_j + T_j)$) A soma dos atrasos e adiantamentos com relação ao tempo devido d_j de se entregar uma tarefa. Para versão de custos, $\sum(EC_j + TC_j)$. Já quando relacionado a uma versão ponderada para manutenções, $\sum(w_m\, E_m + w_m\, T_m)$.

- Carga de trabalho total da máquina ($\sum Wm_j$) O somatório do *makespan* de cada máquina.

- Custo total da manutenção ($\sum CM_m$) Os custos totais relacionados com as manutenções.

Assim como em Geurtsen et al. (2022), no presente trabalho o problema é multiobjetivo quando cada objetivo está dentro de uma função $z(\dots)$. Caso contrário, trata-se de problemas com características e ambientes iguais, mas diferentes objetivos sendo tratados. Por fim, $\mathcal{F}$ representa que vários objetivos diferentes foram abordados no estudo. Para leitores mais interessados em tipos de objetivos, características ou ambiente de máquinas, é possível consultar Pinedo (2016), Leung (2004a) ou ainda T'kindt e Billaut (2006).

Leung (2004a) relata que os pesquisadores inicialmente sentiram uma grande dificuldade de resolver problemas mais elaborados de escalonamento, e a medida que a Teoria da Complexidade foi avançando, descobriu-se que vários dos problemas pertencia à classe de problemas $\mathcal{NP}$-difícil, como o $P2|C_{max}$, o $P2||\sum w_i\, C_i$ e o $J2|p_{ij} = 1|\sum T_i$ (BRUCKER, 2007). Portanto, dado a complexidade destes, para aplicações reais comumente se recorre a métodos heurísticos (TALBI, 2009).

Um dos conceitos que podem ser utilizados para identificar a complexidade de um problema é o de redução. Para 2 problemas de decisão P e Q, pode-se falar que P se reduz à Q quando existe uma função que transforme as entradas de P em entradas válidas de Q (BRUCKER, 2007). Assim, conforme o autor, pode-se recorrer à inferências a partir da redução do problema com complexidade desconhecida para um com complexidade já conhecida. Por exemplo, o $R||C_{max}$ é $\mathcal{NP}$-difícil, visto que pode ser reduzido à $P2||C_{max}$ quando em $R2$ há apenas duas máquinas paralelas idênticas.

Conforme Geurtsen et al. (2022), interconectar decisões dos setores de produção e manutenção pode ser necessário para tornar a programação da produção eficiente. Para isso, segundo os autores, é possível: considerar decisões no escalonamento de qual será a periodicidade das manutenções; estabelecer que certas manutenções devem ocorrer dentro de uma janela de tempo; e ainda considerar RMA, a qual é o foco da pesquisa do presente problema, iniciamente investigado por Barreto et al. (2022). Sendo assim, a próxima seção aborda alguns artigos sobre PEMP com RMA.

2.4.1 Máquinas Paralelas com RMA

Geralmente, os tempos de processamento das atividades dentro de um PPP são considerados constantes. Contudo, existe situações diversas nas quais essa premissa não representa bem

o problema enfrentado. Como exemplo, Alfares, Mohammed e Ghaleb (2021) e Biskup (2008) relatam cenários onde: as pessoas estão aprendendo a realizar as atividades; as máquinas se desgastam consideravelmente, tornando-se menos eficientes; e o tempo de espera da tarefa acaba afetando propriedades físicas que impactam o processamento das tarefas.

Para modelar essas situações, Biskup (2008, p. 318-324) e Hsu et al. (2013, p. 165) detalham possibilidades, algumas expressas a seguir:

- Duração baseada na posição da tarefa (pos) A alteração a uma taxa a na duração p_j da tarefa j é dependente da posição r na qual ela é executada no escalonamento. Dentre as formas de representar: $p_{jr} = p_j + a\,r$, $p_{jr} = p_j\,r^a$.

- Duração baseada no tempo da tarefa ($time$) A uma taxa b, a tarefa j, processada na posição r tem sua duração p_j alterada com base no tempo que a máquina passou processando as demais tarefas anteriores à ela. Como exemplo de modelagem, há $p_{jr} = p_j + b\,t_r$, onde t_r é o tempo que a tarefa alocada em r inicia seu processamento.

Em problemas com RMA, comumente há um desgaste das tarefas relacionadas às formulações abordadas acima, havendo ainda atividades modificadoras de taxa, como manutenções, as quais possuem a capacidade de reverter tal desgaste ocorrido durante o horizonte de planejamento do problema. De acordo com Geurtsen et al. (2022), o PEMP foi o tipo de ambiente de máquina mais abordado na literatura de RMA.

Para obter um melhor entendimento de PEMP com RMA, foram analisadas publicações do tema, podendo serem resumidas ao Quadro 2.1.

Quadro 2.1 – Resumo de problemas da Literatura

Autores	**Representação** $\alpha\|\beta\|\gamma$	**Resolução**
Yang (2011)	$P\|RMA_{j-pos-1}\|\sum Wm_j$ $P\|RMA_{j-pos-o}\|\sum Wm_j$	Tempo polinomial
Yang et al. (2012)	$R\|RMA_{j-pos-o}\|\sum Wm_j$	Tempo polinomial
Hsu et al. (2013)	$R\|RMA_{j-pos-1}\|\sum Wm_j, \sum C_j$ $R\|RMA_{j-time-1}\|\sum Wm_j, \sum C_j$	Tempo polinomial
Lee, Yang e Yang (2013)	$R\|RMA_{j-pos-1}, d_j = d\|\sum(L_j + E_j)$	Tempo polinomial
Zarook e Abedi (2014)	$R\|RMA_{j-pos-o}, d_j\|\sum(LC_j + EC_j + CM_m)$	AG
Alfares, Mohammed e Ghaleb (2021)	$P2\|RMA_{j-pos-o}\|C_{max}$	Heurísticas
Gara-Ali, Finke e Espinouse (2016)	$R\|RMA_{j-pos-o}, RMA_{m-time-o}\|\mathcal{F}$	Tempo polinomial
Cheng, Hsu e Yang (2011)	$R\|RMA_{m-time-1}\|\sum Wm_j, \sum C_j$	Tempo polinomial
Yang e Yang (2013)	$R\|RMA_j\|\sum C_j, \sum Wm_j$	Tempo polinomial
Woo e Kim (2018)	$P\|RMA_{j-time-o}\|C_{max}$	Heurísticas e heightPI
Lu et al. (2018)	$R\|RMA_{j-time-o}, RMA_{m-time-o}, batch\|C_{max}$	Heurísticas
Tavana, Zarook e Santos-Arteaga (2015)	$R\|RMA_{j-pos-o}\|z(\mathcal{F})$	Programação por metas, TOPSIS e *Fuzzy* AHP
Da, Feng e Pan (2016)	$Q\|RMA_{time-o}\|z(C_{max}, CM_m)$	NSGA-II

Dentre os primeiros trabalhos do problema abordado, há Yang (2011), o qual lida com os problemas de minimizar a carga de máquinas paralelas idênticas. Dentre as características do problema, o desgaste é proporcional a posição da tarefa e, ao realizar as manutenções, que

devem obrigatoriamente ocorrerem durante o escalonamento. O estudo é expandido por Yang et al. (2012), ao considerar máquinas não relacionadas e que as manutenções não precisam ocorrer. Ambos os estudos são resolvidos em tempo polinomial.

Já Hsu et al. (2013) analisa um problema semelhante aos já mencionados, com diferentes formas de modelar o desgaste para o problema da carga da máquina e do tempo total, havendo uma manutenção a ser realizada. Todas as variantes tratadas pelos autores podem ser resolvidas em tempo polinomial.

Por sua vez, mantendo a resolução em tempo polinomial, Lee, Yang e Yang (2013) foram os primeiros a tratar do problema no contexto de *just in time*. O objetivo do modelo é minimizar os atrasos e adiantamentos das tarefas, as quais todas possuem uma data comum de entrega, sendo o desgaste dependente da posição. O problema é então expandido por Zarook e Abedi (2014), por permitir mais de uma manutenção e incluir no objetivo a visão de custos de manutenção associados aos custos dos atrasos e adiantamentos. Contudo, ao realizar isso, o problema se torna $\mathcal{NP}$-difícil. Por isso, para sua resolução, os autores recorrem ao AG.

Já Alfares, Mohammed e Ghaleb (2021) trata do problema de minimizar o *makespan* da programação em duas máquinas paralelas idênticas com deterioração dos tempos dependentes da posição. Dado que o problema é considerado $\mathcal{NP}$-difícil, os autores realizam testes com várias heurísticas para duas situações: uma na qual não há manutenção, e a outra onde há manutenções a serem realizadas.

Até o presente momento, todos os artigos aqui mencionados estabelecem que manutenção restaura a máquina para o estado de eficiência do início da programação, não considerando situações como quando cada manutenção restaura a máquina para um estado específico de produtividade que pode ser melhor que o inicial. Isto muda com o trabalho de Gara-Ali, Finke e Espinouse (2016), o qual faz essa generalização para alguns problemas no qual o desgaste é relacionado a posição da tarefa. Além disso, os autores consideram que o tempo de realizar uma manutenção aumenta a medida que é alocada mais a frente na programação. Para sua resolução, os problemas são modelados como um problema de alocação, sendo as variantes tratadas resolvidas em tempo polinomial.

Já para problemas onde não há desgaste nos tempos de processamento, apenas os tempos são condizentes com o estado no qual a máquina se encontra, há os trabalhos de Cheng, Hsu e Yang (2011) e Yang e Yang (2013).

Woo e Kim (2018) resolveram o problema de máquina idênticas com desgaste relativo ao tempo no qual se quer minimizar o *makespan*. Além disso, há manutenções que restauram a máquina para o estado inicial da programação. Dado a complexidade do problema, resolvem por PI, matheuristics e meta-heurísticas.

O problema é expandido por Lu et al. (2018) ao considerar produção em lotes, máquinas não relacionadas e que a duração das manutenções aumentam com base no tempo para iniciá-las. Para resolver, os autores propõe PI para um caso especial do problema, o qual é utilizado em uma *matheuristic*, a qual hibridiza as meta-heurísticas de colônia de abelhas e busca tabu.

Já Tavana, Zarook e Santos-Arteaga (2015) resolveram o problema multiobjetivo de PEMP não relacionadas, com desgaste relativo a posição das tarefas a partir de um procedimento de 3 etapas. Inicialmente utilizaram o fuzzy AHP para a escolha das equipes a realizar a manutenção, e o TOPSIS para transformar um problema de 4 objetivos – custos, makespan, atraso e precocidade – em um problema de programação por metas com 2 objetivos.

Por fim, Barreto et al. (2022) não é o primeiro a abordar PEMP com RMA, ou ainda ao combinar os objetivos *makespan* com custos das manutenções. Da, Feng e Pan (2016) ainda se assemelha a Barreto et al. (2022) por tratar de manutenção com desgaste manutenções obrigatórias que restauram a máquina para estados específicos de produtividade.

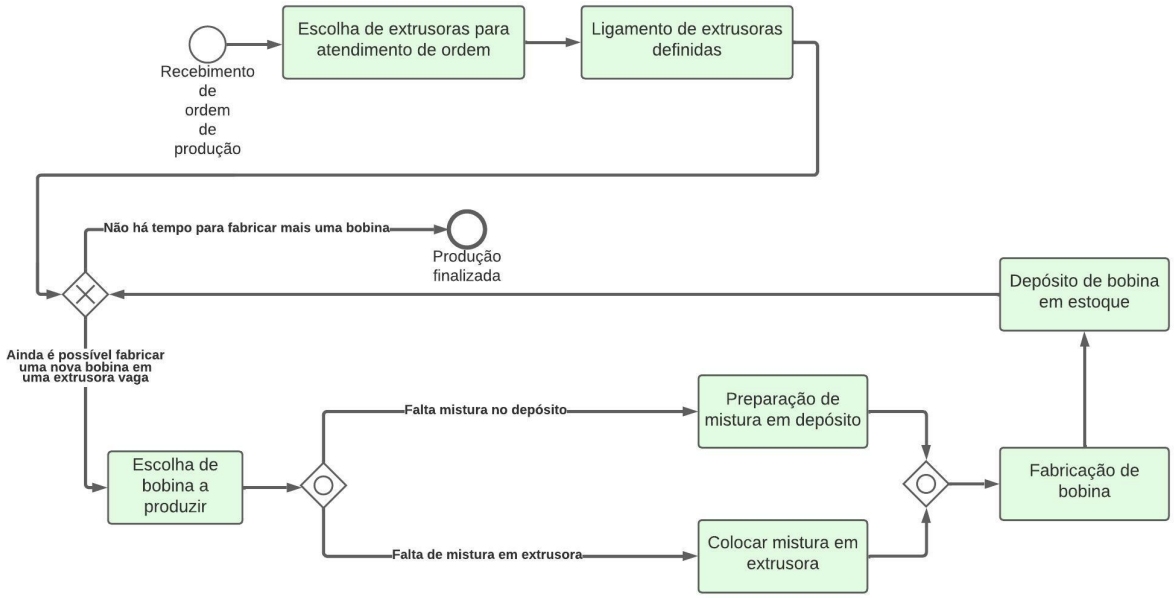
Contudo, a realização de manutenções modificadoras de taxa não afetam apenas os custos das manutenções, pois acabam por influenciar custos na produção por causa da velocidade da máquina, confiabilidade do sistema, dentre outros. Sendo assim, ao tentar minimizar os atrasos e adiantamentos com relação à data devida que deve ser realizada as manutenções, o problema introduzido por Barreto et al. (2022) pode considerar mais fatores relevantes para o problema.

3 UM PROBLEMA DE ESCALONAMENTO DA PRODUÇÃO BI-OBJETIVO COM MÁQUINAS PARALELAS COM ATIVIDADES MODIFICADORAS DE TAXA

3.1 DESCRIÇÃO DO PROBLEMA

O problema em análise foi introduzido por Barreto et al. (2022), o qual foi baseado no processo de produção de bobinas, ilustrado na Figura 6, as quais servem de insumo para produção de sacos e sacolas plásticas recicladas de uma empresa do setor.

Figura 6 – Processo de fabricação de bobinas



O processo de fabricação de bobinas inicia com com o recebimento da ordem de produção do período, sendo definido as máquinas a serem utilizadas para atender essa demanda.

É importante destacar que quando uma máquina é ligada, ela será utilizada durante todo o período, mesmo que para produzir bens fora da ordem de pedidos, dado o alto custo de ligá-la. Sendo assim, enquanto houver tempo de produzir mais uma bobina, a máquina que estiver disponível será utilizada para fabricar uma nova bobina, seja da ordem de produção, ou de escolha da equipe de chão de fábrica, como ilustra a Figura 6.

Para a fabricação de uma bobina, é feita uma mistura de pequenos pedaços de plástico, corante e desumidificador, que é depositada em um recipiente próximo da extrusora, seu próximo destino e onde a mistura será derretida. Esse processo levará a formação do produto semi-final, que será depositado em estoque para posterior uso. Sendo assim, o problema pode ser modelado como PEMP não relacionadas. A reposição de mistura em depósito e na extrusora ocorre regularmente de modo a não deixar faltar material quando necessário.

As máquinas são pré-configuradas nas dimensões da bobina a ser produzida, não havendo na empresa tempos de *setup* dependentes da máquina a qual a bobina será produzida, ou da sequência de tarefas a serem realizadas. Logo, os tempos de *setup* podem ser incluídos nos tempos de processamento das tarefas, sendo executado na etapa de fabricação de bobinas na Figura 6.

Além disso, dado a natureza do processo, cada máquina é submetida a condições extremas como altas temperaturas e pressão, que causam grande desgaste e tornam necessário a realização de manutenções, com o intuito de recuperar suas produtividade ou mesmo reabilitar seu uso. Já o desgaste é proporcional ao tempo que a máquina passou produzindo. Por isso, é necessário modelar o problema com $RMA_{j-tempo-o}$.

A quantidade de manutenções, assim como a sua ordem de execução e sua data de finalização, que podem ser estabelecidos na empresa com base em custos e disponibilidade, são conhecidos *à priori*. Além disso, é importante mencionar que todas as manutenções alocadas no PPP devem ser realizadas, e que elas não necessariamente reparam a máquina para um estado novo.

Por fim, o problema possui uma combinação de objetivos que representam os setores de manutenção e produção da empresa. Para a produção, é importante balancear a produção entre as máquinas, ao mesmo tempo que atender a demanda o mais rápido possível. Já para a manutenção, é vital que as manutenções ocorram próximas ao tempo previamente programado. Sendo assim, o modelo é bi-objetivo e visa minimizar o *makespan* das tarefas, ao mesmo tempo que visa minimizar o somatório dos atrasos e adiantamentos das manutenções.

O presente problema bi-objetivo pode ser representado por $R|RMA_{j-tempo-o}|z(C_{max}, \sum(\alpha\, T_m + \beta\, E_m))$. Como sua versão mono-objetivo com apenas o *makespan* pode ser reduzida à $P2||C_{max}$, o problema em análise pode ser classificado como $\mathcal{NP}$-difícil.

3.2 MODELO MATEMÁTICO

O modelo proposto por Barreto et al. (2022) busca encontrar o conjunto de soluções eficientes do problema bi-objetivo $R|RMA_{j-tempo-o}|z(C_{max}, \sum(\alpha\, T_m + \beta\, E_m))$, onde há n tarefas a serem realizadas. Ele é descrito a seguir.

3.2.1 Conjuntos

- $J = \{1, \ldots, n\}$: Conjunto das n tarefas.
- $H = \{1, \ldots, m\}$: Conjunto de m máquinas.
- $G_h = \{0, 1, \ldots, r_h\}$: Conjunto das r_h manutenções na máquina $h \in H$ e de 0, que representa o seu estado inicial. Sendo assim, G_h representa o conjunto de estados da máquina h.

- $R = \{1, \ldots, n\}$: Conjunto de n posições relativas das tarefas no estado $g \in G_h$ para a máquina h, responsável pela ordenação de tarefas no estado g.

3.2.2 Parâmetros

- p_{jgh}: Tempo de processamento da tarefa $j \in J$ na máquina $h \in H$ após o estado $g \in G_h$.
- d_{gh}: Data de entrega para o estado $g \in G_h$ na máquina $h \in H$.
- t_{gh}: Duração de realização do estado $g \in G_h$ na máquina $h \in H$.
- α_{gh}: Peso de realizar a manutenção $g \in G_h$ atrasada na máquina $h \in H$.
- β_{gh}: Peso de realizar a manutenção $g \in G_h$ adiantada na máquina $h \in H$.

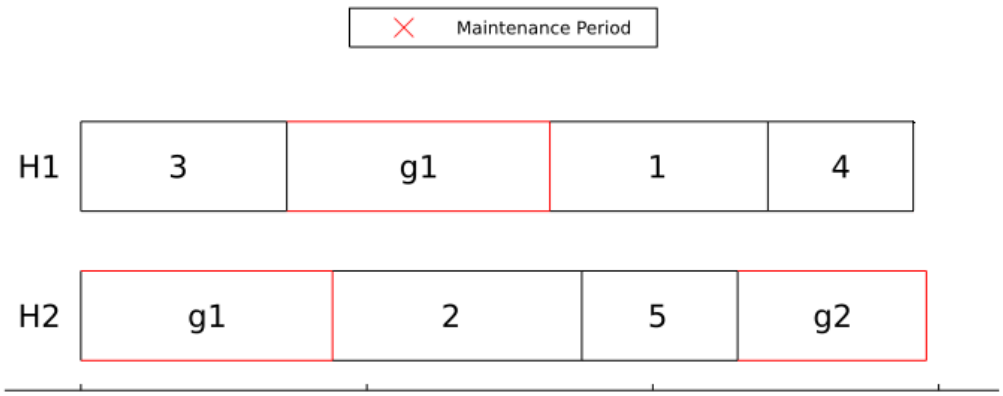
  Ambos os pesos α_{gh} e β_{gh} ficam a critério do responsável pela utilização do modelo.

3.2.3 Variáveis

- x_{jrgh} = Variável binária que assume valor 1 se a tarefa $j \in J$ é realizada na posição $r \in R$ no estado $g \in G_h$ da máquina $h \in H$, ou 0 caso contrário.
- y_{gh} = Variável binária que representa se algum item é processado na máquina $h \in H$ no ou após o estado $g \in G_h$.
- C'_{max} = Variável contínua que representa o *makespan* das tarefas.
- C_{gh} = Variável contínua que representa o tempo de finalização da manutenção $g \in G_h$ na máquina $h \in H$.
- E_{gh} = Variável contínua que representa o adiantamento no tempo da manutenção $g \in G_h$ na máquina $h \in H$.
- T_{gh} = Variável contínua que representa o tempo de atraso da manutenção $g \in G_h$ na máquina $h \in H$.

Para ilustrar o funcionamento da variável x_{jrgh}, a Figura 7 representa uma solução de uma instância qualquer com 5 tarefas, 2 máquinas, nas quais 1 tem apenas uma manutenção a ser realizada, e a outra, 2 manutenções.

Figura 7 – Solução exemplo em gráfico de Gantt



Neste contexto, dentre todas as variáveis x_{jrgh}, apenas x_{3101}, x_{1111}, x_{4211}, x_{2112} e $x_{5,2,1,2}$ possuem valores iguais a 1, as demais variáveis para esta solução possuem valor 0.

3.2.4 Modelo

$$\min Z_2 = C'_{max} \tag{3.1}$$

$$\min Z_1 = \sum_{h \in H} \sum_{g=1}^{r_h} (\alpha_{gh} T_{gh} + \beta_{gh} E_{gh}) \tag{3.2}$$

Sujeito à:

$$\sum_{h\in H}\sum_{g\in G_h}\sum_{r\in R} x_{jrgh} \geq 1 \qquad \forall j \in J \tag{3.3}$$

$$\sum_{j\in J} x_{jrgh} \leq 1 \qquad \forall h \in H, g \in G_h, r \in R \tag{3.4}$$

$$\sum_{j\in J} x_{jrgh} \geq \sum_{j\in J}\sum_{r'=r+1}^{n} x_{jr'gh} \qquad \begin{array}{l}\forall h \in H, g \in G_h,\\ r \in \{1,\ldots,n-1\}\end{array} \tag{3.5}$$

$$y_{g,h} \geq \sum_{g'=g+1}^{r_h} y_{g'h} \qquad \forall h \in H, g \in \{0,\ldots,r_h-1\} \tag{3.6}$$

$$x_{jrgh} \leq y_{gh} \qquad \forall j \in J, r \in R, h \in H, g \in G_h \tag{3.7}$$

$$C_{gh} = \sum_{j\in J}\sum_{g'=0}^{g-1}\sum_{r\in R} p_{jg'h}\, x_{jrg'h} + \sum_{g"=1}^{g} t_{g"h} \qquad \forall h \in H, g \in \{1,\ldots,r_h\} \tag{3.8}$$

$$C_{gh} + E_{gh} - T_{gh} = d_{gh} \qquad \forall h \in H, g \in \{1,\ldots,r_h\} \tag{3.9}$$

$$C'_{max} \geq \sum_{j\in J}\sum_{g\in G_h}\sum_{r\in R} p_{jgh}\, x_{jrgh} + \sum_{g\in G_h} t_{gh}\, y_{gh} \qquad \forall h \in H \tag{3.10}$$

$$x_{jrgh} \in \{0,1\} \qquad \begin{array}{l}\forall j \in J, r \in Rh \in H,\\ g \in G_h\end{array} \tag{3.11}$$

$$y_{gh} \in \{0,1\} \qquad \forall h \in H, g \in G_h \tag{3.12}$$

$$C_{gh}, E_{gh}, T_{gh} \in \mathbb{R}_+ \qquad \forall h \in H, g \in G_h \tag{3.13}$$

A Função Objetivo (3.1) busca minimizar o *makespan* das tarefas, enquanto que a (3.2) visa minimizar os atrasos e adiantamentos das manutenções. As Restrições (3.3) obrigam que todas as tarefas devam ser finalizadas. Já Restrições (3.4) impedem que mais de uma tarefa ocupe a mesma posição no escalonamento. Por sua vez, Restrições (3.5) obrigam que seja mantida a ordem de processamento das tarefas no escalonamento. De forma semelhante, (3.6) faz o controle das atividades de manutenção. Já as Restrições (3.7) relacionam as variáveis x_{jrgh} com y_{gh}. Restrições (3.8) contabilizam o tempo para a finalização de cada manutenção, sendo esse valor utilizado para estabelecer o atraso ou adiantamento total da manutenção nas equações (3.9). Já para controlar o *makespan*, há as Restrições (3.10). Por fim, as Restrições (3.11)-(3.13) determinam o domínio das variáveis.

3.3 MÉTODOS DE RESOLUÇÃO DA LITERATURA

Para lidar com o problema em análise, Barreto et al. (2022) propôs resolvê-lo por 2 métodos distintos, o ε-Restrito e o NSGA-II. Na publicação dos autores, os parâmetros foram definidos de forma arbitrária e com base em experimentação empírica.

3.3.1 Resolução por ε-Restrito

Para resolver pelo ε-Restrito, os autores se basearam no método AUGMECON introduzido por Mavrotas (2009), com o apoio do CPLEX Solver para resolver o modelo apresentado. Como parâmetro do método, há o número de *grid points* que controla a quantidade de problemas a serem resolvidos. Consequentemente, esse parâmetro influencia no tempo gasto para se resolver uma instância, assim como a quantidade de soluções encontradas para ela. No estudo considerada a Equação (3.14).

$$p = \max(50, \frac{n\,m}{2}) \tag{3.14}$$

Por fim, foi definido que o objetivo a ser considerado como restrição no modelo foi o *makespan*, e o tempo de 7 minutos e 30 segundos como limite de tempo para cada iteração do ε-Restrito.

3.3.2 Resolução por NSGA-II

Já para utilizar a heurística baseada no NSGA-II, o qual foi introduzido por Deb et al. (2002), foi necessário inicialmente definir a forma de representar as soluções.

Para a solução apresentada na Figura 7, Barreto et al. (2022) utilizam 3 vetores, como ilustra a Figura 8, onde cada posição de cada vetor representa uma tarefa, e cada valor na posição tem seu significado definido conforme o vetor. Para o, o valor na posição representa a máquina na qual a tarefa será executada; já para v, que tem o objetivo de definir a ordem de execução das tarefas, conta com valores entre 0 e 1, dos quais quanto menor, mais prioritária a tarefa; por último, para w, a quantidade de manutenções que ocorrerá antes da tarefa.

Figura 8 – Solução exemplo em representação por vetores

Tarefa	1	2	3	4	5
Alocação (v)	1	2	1	1	2
Ordem (u)	0,25	0,45	0,01	0,74	0,53
Manutenção (w)	1	1	0	0	0

A partir da representação acima, é possível concluir que, para o planejamento ilustrado, as tarefas 1, 3 e 4 serão processados na máquina $H1$, enquanto que a tarefa 2 e 5, na máquina $H2$. Além disso, as tarefas serão processadas na seguinte ordem: 3, 1 e 4 em $H1$, e 2 e 5 em $H2$. Por fim, serão executadas manutenções antes do processamento das tarefas 1 e 2.

É importante mencionar que $H2$ possui duas manutenções a serem realizadas, mas não há mais que uma manutenção alocada antes das tarefas nessa máquina. Ainda assim, como no problema em análise foi estabelecido que todas as manutenções da instância devem ocorrer dentro do horizonte de planejamento, a manutenção $g2$ de $H2$ deverá ocorrer no final do processamento da máquina, como ilustrado na Figura 7.

Além disso, caso a quantidade de manutenções alocadas em uma máquina pela representação por vetores seja superior a quantidade a ser realizada dentro do horizonte de planejamento, o excedente de manutenções no final do escalonamento, conforme o vetor de ordem, serão ignoradas.

Após isso, os autores estabeleceram os parâmetros utilizados, que estão listados na Tabela 1.

Tabela 1 – Parâmetros do NSGA-II

Parâmetros	**Valor adotado**
Tamanho da população	100
Forma de gerar solução inicial	Solução aleatória
Quantidade de soluções disputando torneio	10
Tipo de cruzamento	Cruzamento uniforme
Tipo de mutação	Mutação de Troca
Taxa de mutação	0.01
Quantidade de iterações	1000

Os parâmetros utilizados por Barreto et al. (2022) são de comum uso na literatura. Para gerar as soluções da população, os valores dos vetores são estabelecidos aleatoriamente utilizando distribuições uniforme entre 0 e 1 para definir a ordem da tarefa, entre 1 e m, para a máquina na qual a tarefa será alocada, e entre 0 e r_h para definir quantas manutenções serão alocadas antes de cada tarefa.

Para expandir a pesquisa de Barreto et al. (2022), no presente trabalho foi estabelecido um novo algoritmo heurístico baseado em BVV para a resolução do problema. Além disso, foi proposta uma heurística construtiva gulosa-aleatória para gerar uma boa solução inicial e, assim, facilitar a busca. Nas próximas seções são apresentados em detalhes o método desenvolvido.

3.4 HEURÍSTICA CONSTRUTIVA GULOSA-ALEATÓRIA

Dado que as heurísticas podem se beneficiar de uma boa solução inicial (GOLDBARG; GOLDBARG; LUNA, 2016), foi proposta uma heurística construtiva gulosa-aleatória dividida em duas etapas e que é detalhada a seguir no Pseudocódigo 3.

Na etapa 1 da heurística, nas Linhas 2 a 5 do Pseudocódigo, é gerada a solução vazia S a partir dos dados da instância. Após isso, no vetor o aloca-se de forma aleatória as tarefas nas máquinas, de forma que a diferença na quantidade de tarefas alocadas em cada máquina seja por no máximo uma. Desse modo, o uso das máquinas é balanceado pela quantidade de tarefas, e assim o objetivo do *makespan* é considerado pela heurística.

A partir disso, são definidos os vetores v e w em uma segunda etapa - operações das Linhas 6 a 15. Dado a ordem estabelecida na execução das manutenções da máquina h, verifica-se qual tarefa alocada em h deve ocorrer antes ou depois de cada manutenção g, de modo a buscar minimizar os atrasos e adiantamentos da manutenção g com relação a data devida d_{gh}.

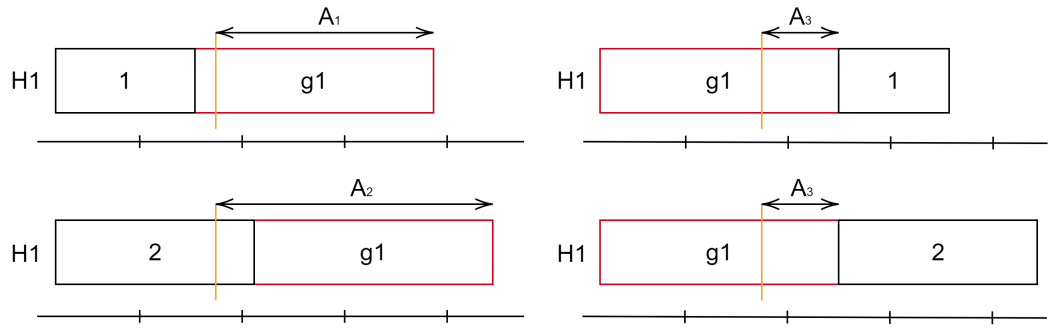
Algoritmo 3: Heurística construtiva gulosa-aleatória

Entrada: Dados de instância
Gerar solução inicial S;
para $j \in J$ **faça**
    # Máquina para a tarefa j definida aleatoriamente do conjunto H
    $S.o_j \leftarrow$ rand(H);
fim
para $h \in H$ **faça**
    $n_h \leftarrow$ Checar_quantidade_de_tarefas_na_maquina_h($S.o$);
    # Vetor de valores de prioridade gerado aleatoriamente em ordem entre 0 e 1 com n_h elementos
    $p \leftarrow$ randsort(n_h);
    $r \leftarrow 1$;
    enquanto $r \leq n_h$ **faça**
        j, manutencao_antes $\leftarrow$ Melhor_tarefa_para_posicao_r(S, h);
        $S.v_j \leftarrow p_r$;
        $r \leftarrow r + 1$; **se** manutencao_antes = 1 **faça** $S.w_j \leftarrow 1$;
    fim
fim
Saída: S.

Para isso, inicia-se verificando a quantidade n_h de tarefas alocadas na máquina h, bem como definindo um vetor de prioridades p de tamanho n_h para definir a ordem das tarefas. Verifica-se então dentre as opções de tarefas disponíveis para serem executadas na posição r da máquina h, qual deve ser processada. Após sua escolha, é modificado os vetores v e w para fixar a ordem de processamento e a posição das manutenções no planejamento. Esse procedimento é realizado para todas as máquinas, de forma a definir a posição de todas as tarefas, assim como manutenções.

Como exemplo dessa operação, considerando que na etapa 1 apenas as tarefas 1 e 2 foram alocadas para a máquina $H1$, a Figura 9 ilustra a opção inicial para $H1$, sendo selecionada a opção com menor A_i. Para as opções abaixo listadas, a escolha será indiferente entre a tarefa 1 e a tarefa 2 ser executada após a realização da manutenção $g1$.

Figura 9 – Opções de Escolha da Segunda Etapa da Heurística Construtiva



Assim, as tarefas são ordenadas, e as manutenções tem o momento de sua execução definido. É importante mencionar que, apesar da problemática abordada permitir que mais de uma manutenção seja executada antes de uma tarefa, esta heurística construtiva permite no máximo apenas uma manutenção antes de uma tarefa.

Esta heurística é utilizada na presente pesquisa para a geração de soluções iniciais para o NSGA-II, assim como para o método Busca em Vizinhança Variável Multiobjetivo, o qual é descrito a seguir.

3.5 BUSCA EM VIZINHANÇA VARIÁVEL MULTIOBJETIVO (BVVMO)

Para a resolução do problema introduzido por Barreto et al. (2022), o presente trabalho se baseou no método da soma ponderada para adaptar o BVV para gerar fronteira de Pareto, com um BVVMO.

3.5.1 Busca em Vizinhança Variável (BVV)

A BVV é uma meta-heurística baseada em estruturas de vizinhança, que foi representado no algoritmo 1 durante a revisão da literatura. Conforme Goldbarg, Goldbarg e Luna (2016), a BVV utiliza diferentes estruturas de vizinhanças para buscar a melhor solução para o problema. Ela se baseia na ideia que diferentes estruturas de vizinhança possuem diferentes mínimos locais, e que os mínimos globais são definidos assim independente da vizinhança utilizada. Abaixo, no Pseudocódigo 4, detalha-se a implementação do presente estudo.

Algoritmo 4: Busca em Vizinhança Variável

Entrada: Dados de instância, solução inicial S_0, peso w_1

$p \leftarrow 0$;
enquanto $p < p_{max}$ **faça**
  $k \leftarrow 1$;
  enquanto $k \leq 2$ **faça**
    $S_1 \leftarrow$ *shaking*$(SN_k(S_0, w_1))$
    para $j = 1$ *para 2* **faça**
      $S_2 \leftarrow$ Busca_local$(LS_j(S_1, w_1))$
      Se $f(S_2, w_1) \leq f(S_1, w_1)$ **Faça** $S_1 \leftarrow S_2$
    fim
    Se $f(S_1, w_1) \leq f(S_0, w_1)$ **Faça** $S_0 \leftarrow S_1$, $k = 1$ e $p = 0$
    Caso contrário **Faça** $k = k + 1$ e $p = p + 1$
  fim
fim

Saída: Melhor solução S_0 encontrada.

Para execução da heurística proposta, é necessário: ler dados da instância; uma solução inicial, que pode ser gerada de qualquer modo; e o peso w_1 do objetivo 1, o qual será posteriormente explanado na equação (3.15).

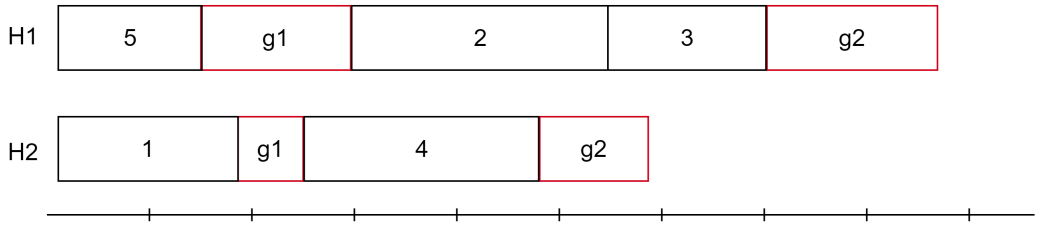
A partir da solução inicial, é aplicado uma operação de *shaking*, que corresponde a uma alteração da solução para uma região de busca diferente a partir de uma estrutura de vizinhança SN_k. Essa etapa é responsável pela diversificação da heurística.

Após isso, são executadas duas operações de busca local, que representam os esforços pela intensificação na heurística. É importante visualizar que a busca muda de solução foco S_0 apenas a partir de uma melhoria clara ou da permanência da qualidade da solução, este último sendo justificado por mudar a região de busca para uma que possa ter soluções melhores.

Considerando a presença de 2 estruturas de vizinhança do tipo *shaking*, quando a estrutura de vizinhança SN_1 não possibilitar que uma solução melhor que S_0 seja encontrada, o tipo de vizinhança para esta etapa mudará para SN_2, sendo a solução S_1 decorrente da aplicação dela submetida a etapa de busca local. Após isso, o BVV obrigatoriamente voltará para a estrutura de vizinhança SN_1, contudo, caso a solução S_2 - encontrada a partir da busca local de S_1 gerada por SN_2 -, seja melhor que S_0, o valor p que representa o critério de parada do método reiniciará. Caso contrário, p terá seu valor aumentado, aproximando-se assim da parada do método.

Antes de detalhar o funcionamento das estruturas de vizinhança utilizadas no estudo, é importante conceituar grupo sob a ótica de tarefas em um PPP considerando RMA. Grupo é conjunto de tarefas que pertencem a mesma máquina, não havendo manutenções alocadas entre elas. Na Figura 10 por exemplo, na máquina $H1$, a tarefa 5 pertence a um grupo, enquanto que as tarefas 2 e 3, a outro, dado que existe a manutenção $g1$ entre os grupos.

Figura 10 – Solução exemplo



Considerando essa classificação, as estruturas de vizinhanças adotadas são apresentadas abaixo, sendo SN_1 e SN_2 utilizadas no *shaking*, e LS_1 e LS_2, na busca local. O funcionamento delas é ilustrado pelas figuras seguintes a partir da modificação da solução apresentada na Figura 10.

- SN_1 Considerando uma mesma máquina, transfere-se a posição de uma tarefa para que, dentre k grupos aleatoriamente selecionados, escolha-se a melhor opção dentre eles considerando o peso w_1. A operação de modificação dessa estrutura de vizinhança pode ser visualizada na Figura 11a, ao trocar da solução ilustrada a tarefa 3 de grupo na máquina $H1$. Essa estrutura de vizinhança possui complexidade de $O(2\,k\,\max_{h \in H}(r_h))$.

- SN_2 Modifica-se de forma aleatória a máquina na qual duas tarefas são selecionadas randomicamente, como ilustra a Figura 11b ao trocar as tarefas 2 e 3 de máquina. Fora

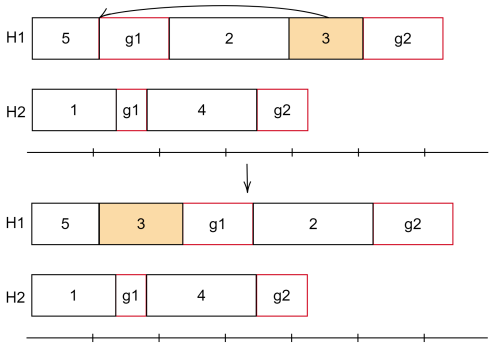
a modificação do vetor v que corresponde a alocação entre máquinas, essa estrutura de vizinhança mantém todas as demais decisões sugeridas pela solução inicial inalteradas. A complexidade de SN_2 é de $O(4\,n\,m)$.

- LS_1 Troca-se o grupo de 2 tarefas de uma mesma máquina, da forma que ilustra a Figura 11c, ao trocar os grupos das tarefas 1 e 4. Essa estrutura de vizinhança seleciona a melhor solução dentre todas as possibilidades de troca. A complexidade dessa estrutura de vizinhança é de $O(2\,n^2 - n)$.

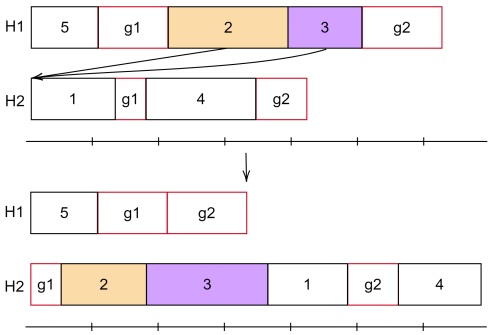
- LS_2 Faz-se a troca de máquinas entre duas tarefas, mantendo a posição original da tarefa trocada, como no caso ilustrado pela Figura 11d, ao trocar as posições entre as tarefas 1 e 5. Assim como em LS_1, seleciona-se a melhor solução dentre todas as opções da vizinhança. Assim como LS_1, a complexidade dessa estrutura é de $O(2\,n^2 - n)$.

Figura 11 – Estruturas de vizinhança

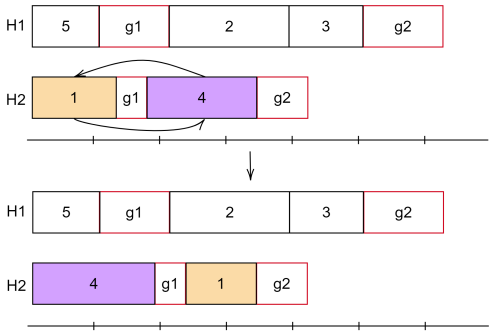
(a) Operação SN_1



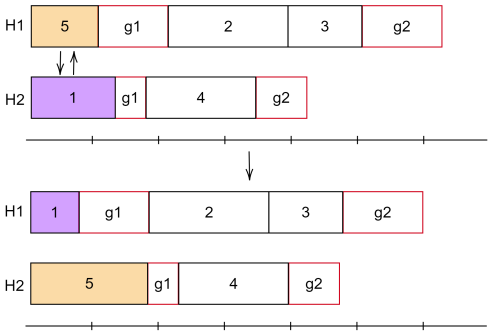
(b) Operação SN_2



(c) Operação LS_1



(d) Operação LS_2



3.5.2 Soma Ponderada

A partir da utilização do BVV, as funções objetivos são somadas de forma ponderada em uma única função, como modelado na equação (3.15). O Pseudocódigo 5 detalha melhor o método.

$$\min w_1 f_1 + (1 - w_1) f_2 \tag{3.15}$$

Algoritmo 5: Método da soma ponderada

Entrada: Dados de instância

1 Gerar intervalo W de pesos de w_1 para resolução de cada iteração, conjunto de soluções não dominadas F;
2 **para** $w_1 \in W$ **faça**
3 | Gerar solução inicial S_0;
4 | $S_1 \leftarrow$ Gerar solução por BVV(S_0, instância, w_1);
5 | $F \leftarrow$ Verificar solução S_1 em F
6 **fim**

Saída: Conjunto de soluções não dominadas F.

Inicialmente é necessário gerar um conjunto de pesos $W = \{w \in \mathbb{R}^+ | w \leq 1\}$ para resolver o problema da soma ponderada apresentado em (3.15). Sendo assim, considerando as limitações do método de soma ponderada para a geração de soluções não dominadas, é o tamanho de W que define quantos problemas e, consequentemente, ajuda a definir quantas soluções serão geradas no máximo para o problema. Na aplicação do estudo, foi definido que W seria o conjunto de 50 pesos igualmente espaçados entre 0 e 1.

A partir disso, entra-se na repetição a qual é responsável por resolver os diferentes problemas para cada conjunto de pesos no intervalo definido W. Começa-se pela geração de soluções iniciais, que no presente trabalho se trata da heurística construtiva gulosa-aleatória que foi descrita anteriormente. Após isso, o problema é resolvido por BVV para a instância considerando o peso w_1.

Após isso, compara-se a solução encontrada S_1 com as demais soluções já pertencentes à fronteira de Pareto encontrada considerando as relações de dominância discutidas durante a revisão da literatura. Desse modo, a fronteira F terá apenas as soluções não dominadas que foram encontradas.

Dessa forma, os resultados obtidos a partir dos métodos AUGMECON e NSGA-II são comparados com os do BVVMO, sendo detalhados a seguir.

4 RESULTADOS COMPUTACIONAIS

As heurísticas propostas foram implementadas na linguagem de programação Julia 1.7.3. Já o modelo, na linguagem de modelagem matemática JuMP, sendo resolvido pelo solver comercial CPLEX 20.1.0.0. Os experimentos foram executados em um computador Intel Core i5 8265U CPU 1.60GHz 1.80 GHz com 16gb de RAM.

4.1 GERAÇÃO DE INSTÂNCIAS

Dado a ausência de problemas com características iguais ao introduzido por Barreto et al. (2022), os autores precisaram gerar instâncias para poder avaliar o desempenho do modelo e da heurística utilizada.

Dentro de um contexto de máquinas uniformes, por ser o da empresa a qual a pesquisa dos autores se baseou, foram geradas instâncias com a quantidade de tarefas $n \in \{5, 10, 15, 20, 25, 30\}$, e de máquinas $m \in \{2, 4, 6\}$. Para cada combinação possível de n por m foram geradas 10 instâncias, totalizando 180. Cada instância é nomeada considerando a quantidade de tarefas e máquinas e uma numeração de 1 a 10, por exemplo, para uma instância com 10 tarefas e 2 máquinas, "10x2_instance_1.xlsx", sendo elas agrupadas a partir de n e m para apresentação dos resultados. Para gerar cada instância, seguiu-se as distribuição dos parâmetros descritas abaixo.

Para cada tarefa j, foi assumido uma carga conforme a distribuição uniforme $U\{2800, 3600\}$. Já para cada máquina h, foi definida uma capacidade inicial de processamento da carga por unidade de tempo de $U\{60, 70\}$, a qual, após as manutenções, pode adquirir uma capacidade a mais de $c_h \in U\{20, 30\}$ após a realização da última manutenção na máquina.

Por sua vez, cada manutenção g que ocorre na máquina h tem duração definida conforme a distribuição uniforme $U\{72, 120\}$, tem a capacidade de recuperar a capacidade da máquina para seu melhor estado estabelecido com $U\{0.8, 1.0\}$. Já a data de entrega devida d_{gh} segue $U\{0, 1\}$ vezes o produto entre o tempo médio de e a quantidade de tarefas n dividido pela quantidade de máquinas m.

Por fim, foi definido que, para as instâncias aqui testadas, os pesos de atraso e adiantamento das manutenções α e β são iguais a 1.

4.2 MÉTRICAS DE PERFORMANCE MULTIOBJETIVO

Para realizar a comparação entre métodos de resolução multiobjetivo, dado a provável ausência de uma solução que domine todas as demais, é necessário adotar algumas métricas relacionadas aos conjuntos de soluções encontrados por cada uma. No presente trabalho, foi adotado as seguintes métricas, utilizadas na pesquisa de Afzalirad e Rezaeian (2017).

- Número de Soluções da Fronteira (NSF): a quantidade de soluções não dominadas que a fronteira possui. Quanto maior a quantidade de soluções, melhor.

- Métrica de Diversificação (MD): essa métrica avalia a amplitude das soluções não dominadas encontradas na fronteira, sendo a fronteira com MD mais alto o melhor conforme essa métrica. É calculado pela seguinte equação:

$$MD = \sqrt{(maxf_1 - minf_1)^2 + (maxf_2 - minf_2)^2} \tag{4.1}$$

- Distância Ideal Média (DMI): calcula-se a distância média das soluções do conjunto não dominado com a solução ideal a partir de $c_i = \sqrt{f_{1i}^2 + f_{2i}^2}$, sendo melhor soluções com DMI menores. Utiliza-se as seguintes equações:

$$DMI = \frac{\sum_{i=1}^{NSF} c_i}{NSF} \tag{4.2}$$

- Dispersão das Soluções Não Dominadas (DSND): mede-se a dispersão das soluções não dominadas de uma fronteira a partir da equação abaixo:

$$DSND = \sqrt{\frac{\sum_{i=1}^{n}(DMI - c_i)^2}{NSF - 1}} \tag{4.3}$$

  Quanto maior o DSND, melhor a fronteira.

- Métrica de Qualidade (MQ): une-se as fronteiras de pareto em um novo conjunto, do qual será eliminadas as soluções dominadas do conjunto, gerando uma nova fronteira de Paretto. Após isso, verifica-se a porcentagem de soluções pertencentes a nova fronteira que fazem parte de cada fronteira, sendo a com a maior porcentagem a melhor fronteira conforme a MQ.

Além dessas métricas, foi coletado o tempo de encontrar a respectiva fronteira de Pareto pelo método de resolução selecionado.

4.3 ANÁLISE DE RESULTADOS

Dado que foram avaliados métodos heurísticos, na presente pesquisa cada método foi executado 4 vezes em cada instância, sendo o valor apresentado em cada dimensão $n \times m$ das tabelas abaixo igual a média do desempenho na métrica em todas as 10 instâncias geradas nas 4 execuções. É ainda marcado em negrito o método com melhor desempenho entre os testados da tabela. As instâncias foram divididas em pequenas, médias e grandes conforme a quantidade de tarefas e a quantidade de máquinas.

Tabela 2 – Comparação entre NSGA-II e BVVMO para instâncias pequenas

	NSF		MD		DMI		DSND		MQ		TIME	
Dimensão	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO
5x2	5,900	5,350	262,426	341,486	0,897	0,839	0,101	0,172	0,546	0,511	1,997	0,083
5x4	4,725	5,650	196,814	481,518	0,928	0,803	0,102	0,202	0,419	0,609	1,948	0,112
5x6	4,500	6,575	245,869	957,048	0,895	0,821	0,110	0,156	0,227	0,786	1,972	0,142
10x2	42,700	7,750	280,684	332,687	0,847	0,753	0,069	0,158	0,815	0,257	2,129	0,268
10x4	7,050	8,425	555,286	644,503	0,836	0,766	0,142	0,204	0,351	0,651	2,029	0,305
10x6	5,450	9,350	470,246	852,461	0,873	0,794	0,133	0,157	0,230	0,776	2,015	0,406
	11,721	7,183	335,221	**601,617**	0,879	**0,796**	0,110	**0,175**	0,431	**0,598**	2,015	**0,219**

Tabela 3 – Comparação entre NSGA-II e BVVMO para instâncias médias

	NSF		MD		DMI		DSND		MQ		TIME	
Dimensão	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO
15x2	58,700	17,125	322,807	627,880	0,848	0,854	0,120	0,099	0,697	0,308	2,362	0,566
15x4	8,150	11,375	743,388	885,146	0,822	0,760	0,164	0,187	0,336	0,664	2,355	0,720
15x6	5,825	9,575	578,450	1137,164	0,844	0,780	0,148	0,183	0,189	0,817	2,243	0,855
20x2	70,925	18,625	806,422	1435,884	0,740	0,609	0,137	0,238	0,762	0,243	2,394	1,010
20x4	20,225	20,750	562,709	836,822	0,814	0,631	0,148	0,248	0,342	0,693	2,399	1,549
20x6	6,200	10,550	753,727	1232,529	0,845	0,780	0,159	0,177	0,232	0,774	2,279	1,916
	28,338	14,667	627,917	**1025,904**	0,819	**0,736**	0,146	**0,189**	0,426	**0,583**	2,339	**1,103**

Tabela 4 – Comparação entre NSGA-II e BVVMO para instâncias grandes

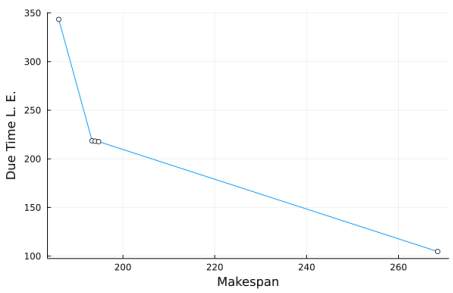
	NSF		MD		DMI		DSND		MQ		TIME	
Dimensão	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO	NSGA II	BVVMO
25x2	83,825	15,250	844,944	2035,428	0,759	0,694	0,113	0,256	0,841	0,164	2,391	1,663
25x4	16,800	24,025	898,247	1229,809	0,764	0,617	0,171	0,214	0,235	0,767	2,468	2,845
25x6	5,825	15,700	344,130	999,802	0,836	0,676	0,157	0,225	0,105	0,902	2,304	3,615
30x2	85,350	15,500	1040,112	2983,948	0,783	0,606	0,111	0,261	0,831	0,176	2,450	2,542
30x4	22,325	26,525	921,569	1790,673	0,742	0,554	0,182	0,230	0,273	0,728	2,516	4,484
30x6	8,350	19,875	640,233	1350,868	0,790	0,588	0,180	0,257	0,147	0,870	2,389	5,569
	37,079	19,479	781,539	**1731,755**	0,779	**0,623**	0,152	**0,241**	0,405	**0,601**	**2,420**	**3,453**

Como é possível verificar, a partir das tabelas acima, a heurística BVVMO tem desempenho médio superior em quase todas as métricas para todas as dimensões, sendo pior apenas na quantidade de soluções encontradas. Dentre outras possíveis questões, isso pode estar evidenciando a incapacidade de métodos de soma ponderada de gerar soluções de áreas não convexas da fronteira de Pareto, como relatado por Marler e Arora (2004) e Ehrgott (2005).

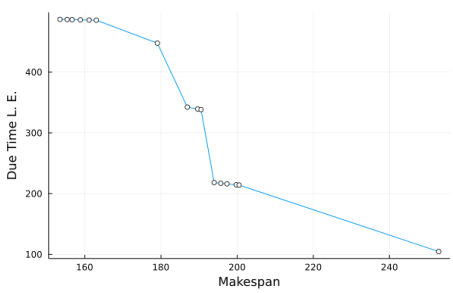
Pode-se verificar isso ao analisar as fronteiras de Pareto de uma instância 5×2 geradas pela BVVMO, representada na Figura 12a, pelo AUGMECON, na Figura 12b.

Figura 12 – Fronteiras de Paretto

(a) Fronteira do BVVMO



(b) Fronteira do AUGMECON



Fora isso, com relação ao tempo de resolução, nota-se uma pouca variação com o crescimento do tamanho do problema para o NSGA-II, enquanto que para o BVVMO é reparado um aumento do tempo a medida que a instância cresce.

Logo, apesar de ter um bom desempenho quando comparado ao NSGA-II implementado por Barreto et al. (2022), nota-se que a BVVMO possui uma desvantagem considerável, e que a adoção de outra forma de lidar com a natureza multiobjetivo do problema pode trazer melhores resultados.

Desconsiderando por esse motivo o desempenho pior em quantidade de soluções geradas, verifica-se uma superioridade total do BVVMO sob o NSGA-II, um evento que não é comum conforme Silberholz et al. (2019), uma meta-heurística ser melhor em todas as métricas, e isso pode ser decorrente da diversidade limitada das instâncias geradas. As instâncias criadas por Barreto et al. (2022) são apenas de máquinas paralelas uniformes, o que pode explicar o desempenho superior da BVVMO já que isso pode representar uma amostra limitada para os demais tipos de situações que os métodos se propõe a solucionar.

Além disso, outro possível motivo que pode explicar o desempenho superior do BVVMO em relação ao NSGA-II é que não foram utilizados métodos de ajustes de parâmetros, como o I-Race, o ParamILS e o Evoca. Advogados por Montero, Riff e Rojas-Morales (2018), os métodos de ajustes de parâmetros podem melhorar o desempenho das heurísticas ao tomar decisões embasadas em dados para utilizar parâmetros. Como não foi utilizado, a escolha dos parâmetros do NSGA-II pode não ter sido a mais adequada que os do BVVMO.

Considerando então essas limitações da pesquisa, sob as condições estabelecidas, o BVVMO é superior ao NSGA-II para a resolução do problema em análise. Contudo, com relação ao AUGMECON, como é possível visualizar pela 5, a BVVMO possui um desempenho consideravelmente baixo pelas métricas aqui adotadas.

Tabela 5 – Comparação entre AUGMECON e BVVMO para instâncias pequenas

	NSF		MD		DMI		DSND		MQ		TIME	
Dimensão	AUGMECON	BVVMO	AUGMECON	BVVMO	AUGMECON	BVVMO	AUGMECON	BVVMO	AUGMECON	BVVMO	AUGMECON	BVVMO
5x2	16,6	5,350	583,681	341,486	0,810	0,839	0,134	0,172	0,853	0,183	4,341	0,083
5x4	24,3	5,650	687,276	481,518	0,812	0,803	0,125	0,202	0,894	0,148	1,882	0,112
5x6	26,2	6,575	941,847	957,048	0,792	0,821	0,117	0,156	0,902	0,123	2,303	0,142
10x2	26,3	7,750	466,257	332,687	0,743	0,753	0,170	0,158	0,847	0,161	115,720	0,268
10x4	38,8	8,425	1173,649	644,503	0,779	0,766	0,131	0,204	0,898	0,111	154,848	0,305
10x6	40,1	9,350	1356,669	852,461	0,781	0,794	0,114	0,157	0,906	0,106	55,160	0,406
	28,717	7,183	**868,230**	**601,617**	**0,786**	**0,796**	0,132	**0,175**	**0,883**	**0,139**	55,709	**0,219**

Logo, apesar de, em comparação com o NSGA-II, e considerando que elas foram submetidas a limitações no estudo, a heurística proposta ter um bom desempenho, ela ainda deve ser melhorada para se aproximar mais de encontrar a fronteira de Pareto do problema. É então promissor o estudo da BVV de base para solucionar o problema multiobjetivo em análise.

É importante ressaltar que foram resolvidas apenas as instâncias pequenas pelo AUGMECON por causa a complexidade do problema e das limitações de tempo do presente estudo, que podem até ser observadas pelo tempo médio de resolução das instâncias pequenas selecionadas.

5 CONCLUSÃO E TRABALHOS FUTUROS

O presente trabalho buscou expandir o estudo de Barreto et al. (2022) para resolver o problema bi-objetivo $R|RMA_{j-tempo-o}|z(C_{max}, \sum(\alpha\, T_m + \beta\, E_m))$, ao propor um novo método de resolver o problema. Para isso, foi necessário realizar um levantamento bibliográfico sobre escalonamento da produção, focando no contexto de manutenções como atividades modificadoras de taxa, bem como de tarefas que sofrem desgaste, onde foi possível perceber a relevância do estudo dos autores ao considerar um problema não visto antes na literatura.

Além disso, foi necessário realizar um estudo sobre meta-heurísticas e otimização multiobjetivo para propor um método de resolução para o problema, bem como entender como realizar a comparação entre métodos de resolução multiobjetivos.

Foi então desenvolvido um método de geração de soluções semi guloso para o problema, de forma que a busca seja iniciada em uma região de busca mais promissora que uma solução gerada aleatoriamente, como realizada por Barreto et al. (2022). Além disso, para encontrar a solução da fronteira de Pareto, foi elaborada uma Busca em Vizinhança Variável Multiobjetivo (BVVMO), baseado no método multiobjetivo de soma ponderada. Ele teve seu desempenho comparado com os métodos NSGA II e AUGMECON que foram utilizados pelos autores, com base em 5 métricas para métodos multiobjetivos.

É importante salientar que para a comparação das heurísticas, foram utilizadas as mesmas instâncias do estudo de Barreto et al. (2022), que corresponde apenas a instâncias de máquinas uniformes, o que, conforme Silberholz et al. (2019), pode afetar a qualidade da comparação. Além disso, não foram adotados métodos de ajuste de parâmetros.

Sendo assim, para as instâncias testadas, nas condições definidas, a heurística BVVMO proposta no estudo, baseada em BVV, possui um desempenho superior ao NSGA II em quase todas as métricas de comparação utilizadas. A única métrica em que o NSGA II possui um desempenho superior a BVVMO é a da quantidade de soluções pertencentes a fronteira de Pareto gerados pelos métodos, o que pode ser explicado pela limitação do método BVVMO discutida no trabalho.

Já com relação a comparação do BVVMO com o AUGMECON, notou-se um baixo desempenho pelas métricas adotadas, mesmo que tendo sido resolvido apenas instâncias pequenas.

Portanto, como sugestões de trabalhos futuros, pode-se:

- Utilizar outros métodos que consigam gerar soluções em regiões não convexas da fronteira, que se baseiem no BVV para resolver o problema multiobjetivo em questão. Dentre as alternativas, pode-se adotar o TPLS ou o *Two Phase Pareto Local Search*;

- Utilizar métodos de ajuste de parâmetros para comparar os métodos com seu desempenho melhorado, que pode explicar o baixo desempenho dos métodos heurísticos. Apos isso,

caso não haja uma melhora significativa nas heurísticas, podem ser adotados esforços para melhorá-las ao:

  - Verificar se as estruturas de vizinhança utilizadas são competentes para as etapas de busca local e *shaking*;
  - Verificar outros operadores de cruzamento e mutação para o NSGA-II.

- Por fim, é importante estabelecer um novo conjunto de instâncias a serem testadas, para que o desempenho dos métodos possa ser comparado em uma amostra mais representativa do tipo de problema que eles foram propostos para resolver.

REFERÊNCIAS

AFZALIRAD, M.; REZAEIAN, J. A realistic variant of bi-objective unrelated parallel machine scheduling problem: Nsga-ii and moaco approaches. **Applied Soft Computing Journal**, Elsevier Ltd, v. 50, p. 109–123, 1 2017.

ALFARES, H.; MOHAMMED, A.; GHALEB, M. Two-machine scheduling with aging effects and variable maintenance activities. **Computers and Industrial Engineering**, Elsevier Ltd, v. 160, 10 2021.

ALMEIDA, A. T. de et al. **Multicriteria and Multiobjective Models for Risk, Reliability and Maintenance Decision Analysis**. [S.l.]: Springer London, Limited, 2015.

ARENALES, M. et al. **Pesquisa Operacional**. Rio de Janeiro: Elsevier: ABEPRO, 2011.

BARRETO, M. et al. Um problema de escalonamento de manutenções e tarefas em máquinas paralelas não relacionadas: aplicação em uma empresa de polímeros reciclados. **Innovation for Systems Information and Decision Meeting**, p. 1, 10 2022.

BISKUP, D. A state-of-the-art review on scheduling with learning effects. **European Journal of Operational Research**, v. 188, p. 315–329, 7 2008.

BRUCKER, P. **Scheduling Algorithms**. New York: Springer, 2007. 371 p.

CHANG, Y.-H.; YEH, C.-H.; SHEN, C.-C. A multiobjective model for passenger train services planning: application to taiwan's high-speed rail line. **Transportation Research Part B: Methodological**, v. 34, n. 2, p. 91–106, 2000. ISSN 0191-2615. Disponível em: https://www.sciencedirect.com/science/article/pii/S0191261599000132.

CHENG, T. C.; HSU, C. J.; YANG, D. L. Unrelated parallel-machine scheduling with deteriorating maintenance activities. **Computers and Industrial Engineering**, v. 60, p. 602–605, 5 2011.

CHENG, T. C.; SIN, C. C. A state-of-the-art review of parallel-machine scheduling research. **European Journal of Operational Research**, v. 47, p. 271–292, 1990.

CORRêA, H. L.; GIANESI, I. G. N.; CAON, M. **Planejamento, Programação e Controle da Produção - MRP II / ERP**. 8. ed. São Paulo: Atlas, 2019.

COSTA, L. C. A. D. **Uma heurística baseada em busca local de Pareto para o Pollution-Rounting Problem bi-objetivo**. Dissertação (Mestrado) — Universidade Federal da Paraíba, 2015.

DA, W.; FENG, H.; PAN, E. Integrated preventive maintenance and production scheduling optimization on uniform parallel machines with deterioration effect. In: **2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)**. [S.l.: s.n.], 2016. p. 951–955.

DEB, K. Multi-objective optimisation using evolutionary algorithms: an introduction. In: **Multi-objective evolutionary optimisation for product design and manufacturing**. [S.l.]: Springer, 2011. p. 3–34.

DEB, K. et al. **A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II**. 2002.

DENNIS, P. **Produção Lean Simplificada**: Um guia para entender o sistema de produção mais poderoso do mundo. 2. ed. Porto Alegre: Bookman, 2008.

EHRGOTT, M. **Multicriteria Optimization**. 2. ed. Berlin: Springer, 2005. 323 p.

GARA-ALI, A.; FINKE, G.; ESPINOUSE, M. L. Parallel-machine scheduling with maintenance: Praising the assignment problem. **European Journal of Operational Research**, Elsevier B.V., v. 252, p. 90–97, 7 2016.

GEURTSEN, M. et al. **Production, maintenance and resource scheduling: A review**. [S.l.]: Elsevier B.V., 2022.

GOLDBARG, M. C.; GOLDBARG, E. G.; LUNA, H. P. L. **Otimização Combinatória e Meta-Heurísticas**: Algoritmos e aplicações. [S.l.]: Campus, 2016.

HILLIER, F. S.; LIEBERMAN, G. J. **Introdução à Pesquisa Operacional**. 8. ed. São Paulo: McGraw-Hill, 2006.

HSU, C. J. et al. Unrelated parallel-machine scheduling problems with aging effects and deteriorating maintenance activities. **Information Sciences**, v. 253, p. 163–169, 12 2013.

LEE, H. T.; YANG, D. L.; YANG, S. J. Multi-machine scheduling with deterioration effects and maintenance activities for minimizing the total earliness and tardiness costs. **International Journal of Advanced Manufacturing Technology**, v. 66, p. 547–554, 4 2013.

LEUNG, J. Y. T. Introduction and notation. In: LEUNG, J. Y. T. (Ed.). **Handbook of Scheduling**: Algorithms, models, and performance analysis. [S.l.]: Chapman Hall/CRC, 2004. p. 1224.

LEUNG, J. Y. T. A tutorial on complexity. In: LEUNG, J. Y. T. (Ed.). **Handbook of Scheduling**: Algorithms, models, and performance analysis. [S.l.]: Chapman Hall/CRC, 2004. p. 1224.

LU, S. et al. A hybrid abc-ts algorithm for the unrelated parallel-batching machines scheduling problem with deteriorating jobs and maintenance activity. **Applied Soft Computing Journal**, Elsevier Ltd, v. 66, p. 168–182, 5 2018.

MARLER, R. T.; ARORA, J. S. Survey of multi-objective optimization methods for engineering. **Structural and Multidisciplinary Optimization**, v. 26, p. 369–395, 4 2004.

MAVROTAS, G. Effective implementation of the -constraint method in multi-objective mathematical programming problems. **Applied Mathematics and Computation**, Elsevier Inc., v. 213, p. 455–465, 2009.

MIETTINEN, K. Interactive nonlinear multiobjective procedures: State of the art annotated bibliographic surveys. In: EHRGOTT, M.; GANDIBLEUX, X. (Ed.). **Multiple Criteria Optimization State of the Art Annotated Bibliographic Surveys**. London: Springer London, Limited, 2003.

MONTERO, E.; RIFF, M. C.; ROJAS-MORALES, N. Tuners review: How crucial are set-up values to find effective parameter values? **Engineering Applications of Artificial Intelligence**, v. 76, p. 108–118, 2018.

MORABITO, R. Pesquisa operacional. In: BATALHA, M. O. (Ed.). **Introdução à engenharia de produção**. 7. ed. Rio de Janeiro: Elsevier, 2008. p. 157–182.

MULUK, A.; AKPOLAT, H.; XU, J. Scheduling problems — an overview. **Journal of Systems Science and Systems Engineering**, Springer Science and Business Media LLC, v. 12, n. 4, p. 481–492, dec 2003.

PAQUETE, L.; STÜTZLE, T. A two-phase local search for the biobjective traveling salesman problem. In: FONSECA, C. M. et al. (Ed.). **Evolutionary Multi-Criterion Optimization**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003. p. 479–493.

PINEDO, M. L. **Scheduling**: Theory, algorithms, and systems. 5. ed. Lodon: Springer London, Limited, 2016.

SILBERHOLZ, J. et al. Computational comparison of metaheuristics. In: GENDREAU, M.; POTVIN, J.-Y. (Ed.). **Handbook of Metaheuristics**. 3. ed. [S.l.]: Springer, 2019. v. 272, p. 581–604.

TAHA, H. A. **Pesquisa Operacional**: Uma visão geral. 8. ed. São Paulo: Pearson Prentice Hall, 2008.

TALBI, E. G. **Metaheuristics**: From design to implementation. Hoboken: Wiley, 2009. (Wiley Series on Parallel and Distributed Computing).

TAVANA, M.; ZAROOK, Y.; SANTOS-ARTEAGA, F. J. An integrated three-stage maintenance scheduling model for unrelated parallel machines with aging effect and multi-maintenance activities. **Computers and Industrial Engineering**, Elsevier Ltd, v. 83, p. 226–236, 2015.

T'KINDT, V.; BILLAUT, J.-C. **Multicriteria Scheduling**: Theory, models and algorithms. 2. ed. New York: Springer, 2006. 359 p.

T'KINDT, V.; BILLAUT, J.-C.; PROUST, C. Solving a bicriteria scheduling problem on unrelated parallel machines occurring in the glass bottle industry. **European Journal of Operational Research**, v. 135, n. 1, p. 42–49, 2001. ISSN 0377-2217. Disponível em: https://www.sciencedirect.com/science/article/pii/S0377221700002885.

WHITLEY, D. Next generation genetic algorithms: A user's guide and tutorial. In: GENDREAU, M.; POTVIN, J.-Y. (Ed.). **Handbook of Metaheuristics**. 3. ed. [S.l.]: Springer, 2019. v. 272, p. 245–274.

WOO, Y.-B.; KIM, B. S. Matheuristic approaches for parallel machine scheduling problem with time-dependent deterioration and multiple rate-modifying activities. **Computers Operations Research**, v. 95, p. 97–112, 2018.

YANG, D. L. et al. Unrelated parallel-machine scheduling with aging effects and multi-maintenance activities. **Computers and Operations Research**, v. 39, p. 1458–1464, 7 2012.

YANG, D.-L.; YANG, S.-J. Unrelated parallel-machine scheduling problems with multiple rate-modifying activities. **Information Sciences**, v. 235, p. 280–286, 2013. Data-based Control, Decision, Scheduling and Fault Diagnostics.

YANG, S. J. Parallel machines scheduling with simultaneous considerations of position-dependent deterioration effects and maintenance activities. **Journal of the Chinese Institute of Industrial Engineers**, v. 28, p. 270–280, 6 2011.

ZAROOK, Y.; ABEDI, M. Jit-scheduling in unrelated parallel-machine environment with aging effect and multi-maintenance activities. **Int. J. Services and Operations Management**, v. 18, p. 99–113, 2014.

ÓLAFSSON, S. Chapter 21 metaheuristics. In: HENDERSON, S. G.; NELSON, B. L. (Ed.). **Simulation**. [S.l.]: Elsevier, 2006, (Handbooks in Operations Research and Management Science, v. 13). p. 633–654.