UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO DE JOINVILLE
CURSO DE ENGENHARIA DE TRANSPORTES E LOGÍSTICA

ROBERTO CARLOS VIEIRA

SCHEDULING DE ROBÔS MÓVEIS AUTÔNOMOS EM AMBIENTE DE MANUFATURA FLEXÍVEL COM USO DE META-HEURÍSTICAS DE BUSCA EM VIZINHANÇAS

Joinville
2022

ROBERTO CARLOS VIEIRA

SCHEDULING DE ROBÔS MÓVEIS AUTÔNOMOS EM AMBIENTE DE MANUFATURA FLEXÍVEL COM USO DE META-HEURÍSTICAS DE BUSCA EM VIZINHANÇAS

Trabalho apresentado como requisito para obtenção do título de bacharel no Curso de Graduação em Engenharia Transportes e Logística do Centro Tecnológico de Joinville da Universidade Federal de Santa Catarina.

Orientadora: Dra. Silvia Lopes de Sena Taglialenha.

Joinville

2022

ROBERTO CARLOS VIEIRA

SCHEDULING DE ROBÔS MÓVEIS AUTÔNOMOS EM AMBIENTE DE MANUFATURA FLEXÍVEL COM USO DE META-HEURÍSTICAS DE BUSCA EM VIZINHANÇAS

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do título de bacharel em Engenharia de Transportes e Logística, na Universidade Federal de Santa Catarina, Centro Tecnológico de Joinville.

Joinville (SC), 07 de dezembro de 2022.

Banca Examinadora:

_____

Dra. Silvia Lopes de Sena Taglialenha
Orientadora/Presidente

_____

Dra. Christiane Wenck Nogueira Fernandes
Membro (a)
Universidade Federal de Santa Catarina

_____

Dra. Vanina Macowski Durski Silva
Membro (a)
Universidade Federal de Santa Catarina

Dedico este trabalho aos meus pais, pois graças a seus esforços e sacrifícios que hoje posso concluir o meu curso.

AGRADECIMENTOS

A Deus pela família que tenho e pelas experiências de vida pelas quais eu pude me desenvolver e aprender.

A minha família pelos ensinamentos que me fizeram me dedicar cada vez mais aos meus objetivos, e pelo apoio em cada etapa de minha vida até chegar nesse grande dia.

A minha orientadora Silvia Lopes de Sena Taglialenha, por ter visto potencial e acreditado em mim no projeto de iniciação científica que é tema deste trabalho, e ter me ajudado com todo seu conhecimento e profissionalismo.

Ao CNPq por possibilitar ampliar meu conhecimento científico e o desenvolvimento deste trabalho.

Aos meus amigos, colegas de graduação, monitores de disciplinas e professores, pelos apoios e ensinamentos durante todo período de graduação.

RESUMO

A manufatura moderna com flexibilidade de processos introduziu os robôs móveis autônomos (AMR – *Autonomous Mobile Rob*ot) em operações intralogísticas, como manufatura, armazenamento, *cross-docks*, terminais e hospitais. A utilização de tecnologias com o propósito de utilizar de maneira eficiente os seus recursos, trará uma vantagem competitiva com relação ao mercado global. Visando alcançar seus objetivos as empresas tem adotado a mudança para um sistema de manufatura flexível (FMS - *flexible manufacturing system*), junto aos robôs móveis autônomos, tal mudança permite o sistema adaptar-se às frequentes mudanças nas demandas ao longo dos anos. Considerando isso, este trabalho apresenta o problema de sequenciamento de tarefas de dois robôs móveis autônomos, que tem, como objetivo, abastecer quatro alimentadores de máquinas em uma linha de produção localizada em um ambiente de manufatura flexível. O problema consiste em programar os AMRs de forma que as máquinas não interrompam o funcionamento devido à falta de peças. O modelo proposto tem como objetivo minimizar o *makespan* – tempo total para processar todas as tarefas – em uma linha de produção com um horizonte de planejamento pré-estabelecido. O estudo leva em consideração as características das máquinas, do robô e define uma janela de tempo restrita para a realização das tarefas. A característica do problema é *NP-Hard*, portanto, são apresentados dois métodos computacionais baseados em meta-heurísticas de busca em vizinhanças. Para os dados considerados, a meta-heurística baseada em VND tende a encontrar uma solução de boa qualidade para o problema de *scheduling,* enquanto que a meta-heurística baseada em VNS tende a encontrar uma solução de boa qualidade com mais dificuldade, resultando em uma diferença média de 1,25%, entre os métodos de solução propostos, com relação ao cenário base estudado.

Palavras-chave: *Scheduling*. *Autonomous Mobile Rob*ot. *Makespan*. *Variable Neighborhood Descent*. Ambiente de manufatura flexível.

ABSTRACT

Modern manufacturing with process flexibility has introduced Autonomous Mobile Robots (AMR) into intralogistics operations such as manufacturing, warehousing, cross-docks, terminals and hospitals. The use of technologies in order to efficiently use its resources will bring a competitive advantage in relation to the global market. Aiming to achieve their goals, companies have adopted the change to a flexible manufacturing system (FMS - flexible manufacturing system), along with autonomous mobile robots, such a change allows the system to adapt to frequent changes in demands over the years. Considering this, this work presents the task sequencing problem of two autonomous mobile robots, which aims to supply four machine feeders in a production line located in a flexible manufacturing environment. The problem is to program the AMRs so that the machines do not stop working due to lack of parts. The proposed model aims to minimize the makespan – total time to process all tasks – in a production line with a pre-established planning horizon. The study takes into account the characteristics of the machines, the robot and defines a restricted time window for carrying out the tasks. The characteristic of the problem is NP-Hard, therefore, two computational methods based on neighborhood search metaheuristics are presented. For the considered data, the VND-based metaheuristic tends to find a good quality solution to the scheduling problem, while the VNS-based metaheuristic tends to find a good quality solution with more difficulty, resulting in a average difference of 1,25% between the proposed solution methods, in relation to the studied base scenario.

Keywords: *Scheduling*. Autonomous Mobile Robot. Makespan. Variable Neighborhood Descent. Flexible manufacturing environment.

LISTA DE FIGURAS

LISTA DE ALGORITMOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AGV - *Automated Guided Vehicle*

AMPL – *A Mathematical Programming Language*

AMR – *Autonomous Mobile Robot*

FMS - *Flexible Manufacturing System*

FO – Função Objetivo

HSI - Heurística de Solução inicial

PLIM – Programação Linear Inteira Mista

PO – Pesquisa Operacional

SAMR – *Scheduling a Single AMR*

SLC – *Small Loaded Carries*

UGV - *Unmanned Ground Vehicles*

VND - *Variable Neighborhood Descent*

VNS - *Variable Neighborhood Search*

VND-AMR - *Variable Neighborhood Descent* para o AMR

VNS-AMR - *Variable Neighborhood Search* para o AMR

SUMÁRIO

1. INTRODUÇÃO

O primeiro Veículo Guiado Automatizado (AGV - *Autonomous Guided Vehicle*) conhecido foi introduzido pela Barret Electronics de Northbrook, Illinois, EUA, em 1953, e desde então, os AGVs têm sido amplamente utilizados para simplificar a intralogística e os processos de manuseio de materiais em ambientes industriais (ULLRICH, 2014). Nos tempos atuais da produção industrial moderna, a utilização de robôs em conjunto com tarefas humanas está crescendo cada vez mais em espaços compartilhados. Nas últimas décadas, os veículos guiados automatizados (AGV) ou, em certos casos, por robôs móveis automatizados (AMR - *Autonomous mobile robot*) possibilitam maior flexibilidade em ambientes de manufatura. Os sistemas AMR e AGV tornaram-se, assim, fundamentais para os paradigmas de produção modernos, onde o aumento da flexibilidade nas aplicações logísticas é possibilitado por essas tecnologias (FAZLOLLAHTABAR; SAIDI-MEHRABAD, 2015).

Saber reagir corretamente nos sistemas de manufatura a essas mudanças é de suma importância, logo ter um sistema flexível se torna muito valioso. Um sistema de manufatura flexível (FMS - *flexible manufacturing system*) consegue desempenhar papel significativo, principalmente na era da indústria 4.0 (LU, 2017; MACDOUGALL, 2014; THOBEN; WIESNER; WUEST, 2017). Buscando reunir eficiência em grande escala nos sistemas de produção automatizados em que existe a possibilidade de flexibilidade dos processos manuais, as empresas têm utilizado tecnologias de automação sob a supervisão humana e robôs móveis autônomos (AMR) (DANG et al., 2013). As operações que envolvem movimentação de materiais em um FMS são feitas geralmente por veículos guiados automatizados (AGV) ou, em certos casos, por robôs móveis automatizados (AMR - *Automated mobile robot*), o quais podem ou não possuir braços montados em plataformas móveis.

Os AGVs não têm inteligência própria e precisam de trilhos ou fitas magnéticas para se guiar, e são frequentemente usados em aplicações industriais para mover materiais ao redor do chão de fábrica ou em um armazém. Já os AMRs são inteligentes, com capacidade de se locomover nas fábricas e criar as próprias rotas

com um sistema semelhante ao GPS de um carro e, definindo um ponto de saída e de destino, os AMRs, utilizando seu software de mapas gerados, conseguem traçar a melhor rota, geralmente são usados para indicar veículos de manuseio de materiais que podem navegar autonomamente de um lugar para outro para realizar tarefas específicas.

Buscando reduzir custos operacionais e otimizar operações, é viável que as empresas procurem melhorar os processos na cadeia produtiva, aplicando os recursos de forma eficiente, aumentando o nível das operações, ou seja, realizando os processos produtivos em menor tempo e com qualidade. Existem diversas técnicas de otimização empregadas nas indústrias com o objetivo de reduzir os custos das empresas e também de maximizar o uso das máquinas nas tarefas e processos produtivos (BOWERSOX, 2011).

O problema de programação da produção tem como objetivo otimizar os processos de produção, atribuindo de modo eficiente os recursos, levando em consideração as restrições operacionais da empresa, como por exemplo, restrições de processos de fabricação e de capacidade (SHEN et al., 2006). O uso eficiente e otimizado dos recursos na indústria está se tornando fundamental, visto que a melhora na qualidade do serviço, dos processos e na redução de custos, são elementos importantes de uma empresa para a consolidação no mercado.

Uma etapa da programação da produção consiste em realizar o sequenciamento de tarefas (*scheduling*), que, segundo Wight (1984), visa reduzir o tempo de processamento total em cada máquina, garantindo a alocação eficiente dos recursos necessários. De uma forma mais geral, o problema de sequenciamento consiste em um processo de tomada de decisão fundamental para as empresas, e consiste em designar um conjunto de tarefas e um conjunto de máquinas com o objetivo de otimizar uma medida de desempenho como por exemplo, tempo de atraso, tempo de adiantamento, tempo de processo, entre outros.

É comum utilizar o conceito de *scheduling* junto aos AMRs, designando em que máquinas e quais AMRs determinadas tarefas serão executadas, e em que sequência e instante de tempo devem ser iniciadas e finalizadas. (DANG ET AL., 2013; FERNANDES, 2021; RIAZI e LENNARTSON, 2021)

Uma programação eficiente faz com que todas as tarefas sejam executadas em tempo mínimo, ou ainda, define qual a sequência de realização das tarefas para

que o tempo total de execução das mesmas seja o menor possível, e atendendo às restrições do problema (FERNANDES, 2021).

De acordo com Fernandes (2021), a programação das tarefas é decisiva para uma produção eficiente, visto que está relacionada ao planejamento estratégico da empresa e por propiciar a redução dos estoques, cumprindo as datas de entrega e resolvendo gargalos do sistema. Este problema de sequenciamento de multitarefas em máquinas e robôs tem semelhanças com o problema do agendamento simultâneo de máquinas, pois seus subproblemas, o escalonamento de máquinas e o agendamento de AMRs, coincidem com o conjunto dos problemas NP-Difíceis, isto é, são os problemas para os quais não existem algoritmos polinomiais que resultem em soluções ótimas (FISCHETTI; MARTELLO; TOTH, 1987). Nesse caso, é imprescindível a utilização de métodos heurísticos para encontrar uma solução satisfatória para problemas de grande porte (DANG; NGUYEN; RUDOVÁ, 2019; GOLDBERG, 1989; NIELSEN et al., 2014).

Neste contexto, neste trabalho considera-se o problema de sequenciamento de tarefas e robôs móveis autônomos em um ambiente de manufatura flexível. Destacando que o modelo pode ser empregado por qualquer veículo de transporte em que haja movimentação de peças. Tem como base o modelo descrito por Dang et al. (2013), e propõe-se um método computacional baseado em meta-heurísticas de busca em vizinhanças com algumas premissas além do que foi descrito por Dang et al. (2013).

1.1. OBJETIVOS

Para resolver a problemática do sequenciamento e agendamento de tarefas de dois AMR em um ambiente de manufatura flexível, propõe-se neste trabalho os seguintes objetivos.

1.1.1. Objetivo geral

Desenvolver as meta-heurísticas baseadas em *Variable Neighborhood Descent* (VND-AMR) e *Variable Neighborhood Search* (VNS-AMR) para tratar o problema de *scheduling* com dois robôs em ambiente de manufatura flexível.

1.1.2. Objetivos específicos

- Aprofundar os conhecimentos referentes ao tema sequenciamento de tarefas;
- Identificar métodos de busca local baseadas em vizinhança;
- Determinar as restrições necessárias para a solução do problema de *scheduling* utilizando dois robôs, considerando a capacidade dos robôs e janela de tempo;
- Definir a sequência de tarefas a serem realizadas de modo a minimizar o *makespan*;
- Analisar os resultados obtidos ao se alterar a quantidade de SLCs transportados pelos AMRs e da janela de tempo em relação ao cálculo do *makespan*;
- Fazer um comparativo entre os resultados da VND-AMR (*Variable Neighborhood Descent* para o AMR) e VNS-AMR *(Variable Neighborhood Search* para o AMR).

1.2. METODOLOGIA

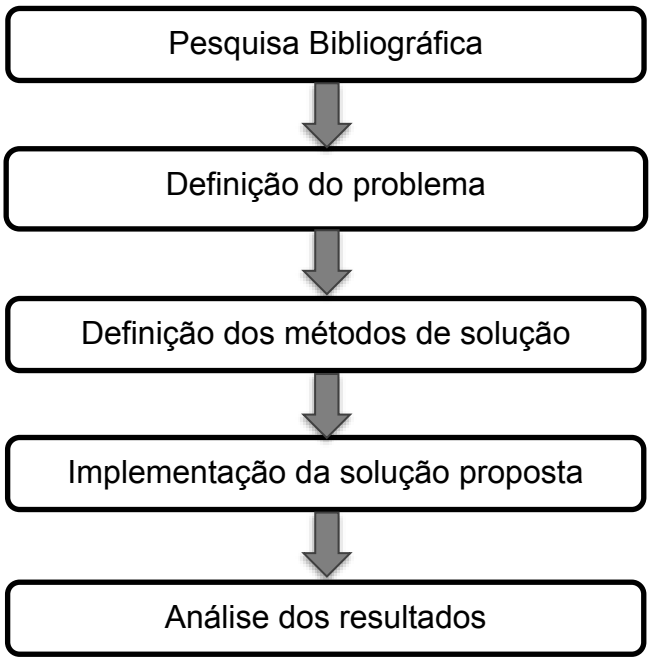
Buscando alcançar os objetivos propostos neste trabalho, planejou-se uma metodologia de trabalho que pode ser considerada, pela sua natureza, como pesquisa aplicada. Segundo Gil (2019), a pesquisa aplicada tem como finalidade a aplicação, utilização e resultados práticos dos conhecimentos, e consiste em determinar solucionar um caso de estudo, identificar as variáveis que podem influenciar nos resultados e definir os métodos de controle e solução. Na Figura 1 ilustram-se as etapas da metodologia considerada.

Primeiramente é realizada a pesquisa bibliográfica referente aos temas que construíram a fundamentação teórica do trabalho, como por exemplo, o problema de *scheduling*, robôs móveis, modelos de sequenciamento de tarefas, métodos heurísticos e meta-heurísticos, entre outros. A pesquisa foi realizada de forma online na biblioteca universitária da Universidade Federal de Santa Catarina (UFSC) e no Google Acadêmico, englobando os idiomas português e inglês, organizado por relevância sem data específica, exceto nos casos práticos, que foi utilizado os últimos cinco anos.

Na etapa seguinte, apresenta-se o detalhamento e a definição do problema de sequenciamento de tarefas, de robôs, e a formulação do modelo matemático. Na sequência, detalha-se o método de solução proposto, o qual é baseado em meta-heurísticas de busca em vizinhanças e implementado em linguagem Python no *software* VSCode, utilizando o sistema operacional Windows 10.

Com a obtenção dos resultados, a etapa seguinte traz uma comparação com os resultados obtidos entre as meta-heurísticas propostas.

Figura 1 - Etapas da metodologia



Fonte: Autor (2022).

1.3. ESTRUTURA DO TRABALHO

Este trabalho está organizado em cinco capítulos sequenciamento estruturado para a melhor compreensão dos temas.

No capítulo 2 detalhe-se os conceitos e características do problema de sequenciamento (*scheduling*), do sequenciamento de máquinas e seus modelos, do sequenciamento de tarefas e robôs autônomos com sua modelagem matemática e complexidade. Ilustra-se também os métodos de solução via modelagem matemática e meta-heurística com exemplos de aplicações.

No capítulo 3 apresenta-se os métodos de solução propostos, sendo um baseado na meta-heurística *Variable Neighborhood Descent* e outra na *Variable Neighborhood Search*. É demonstrado as etapas para obter a solução desde a codificação até a análise da factibilidade.

No capítulo 4 ilustra-se a descrição dos dados utilizados no trabalho e os resultados obtidos com suas análises.

As considerações finais e conclusões obtidas dos métodos de solução estão no capítulo 5.

2. FUNDAMENTAÇÃO TEÓRICA

Apresentam-se neste capítulo os conceitos base no desenvolvimento do trabalho, tais como o problema de sequenciamento de tarefas, sequenciamento de máquinas e seus modelos, definição do problema e métodos de solução.

2.1. PROBLEMA DE SEQUENCIAMENTO

O problema de sequenciamento (*scheduling*) é um processo de otimização dos recursos de máquinas, ferramentas e mão de obra. Segundo Baker (1974), o problema de sequenciamento tem como finalidade aproveitar o máximo de recursos possíveis em um tempo determinado para a execução de um conjunto de processos a serem executados.

De acordo com Reis, J. (2006), de forma geral, assume-se nos problemas de *scheduling* a necessidade de processar ou executar um determinado número de tarefas ou operações, identificando a sequência de execução de operações de acordo com uma ordem determinada, no qual cada processamento necessita de uma máquina ou de um operador durante um intervalo de tempo definido e cada máquina ou operador só pode realizar uma operação por vez.

Veríssimo (2016) cita alguns conceitos do problema de sequenciamento:

- Tempo de processamento: Período em que o a máquina ou posto de trabalho leva para realizar uma operação;
- Tempo de *setup*: Tempo de preparação ou adaptação da máquina a uma nova operação;
- Tempo de conclusão (*makespan*): Momento em que todas as tarefas são processadas;
- Restrição: Todas as situações que limitam a possibilidade de escolha;
- Produto: Resultado de um processo fabril;

- Prioridade/Peso: Característica do produto que determinará a prioridade de sua escolha perante a outro produto;
- Data de disponibilidade: Período no qual é possível realizar um processamento em uma máquina;
- Recursos: Máquinas, matérias primas, mão de obra, ou qualquer outro elemento que seja necessário para a produção;
- Tarefa (*job*): Conjunto de operações a serem realizadas ou processadas.

Para Branquinho (2013), a conclusão de uma tarefa necessita da realização de várias operações a serem processadas em uma determinada ordem, assim, define-se o problema de sequenciamento como um problema de agendamento de operações, em um determinado número de máquinas com suas respectivas características, como, tempos de processamento e tempos de setup, visando buscar o menor tempo possível da execução total de todas as tarefas (*makespan*).

Segundo Pape (2015) o problema de *scheduling* pode ser dividido da seguinte maneira:

- Designação: Define *quem* irá realizar cada tarefa;
- Sequenciamento: Define qual será a *ordem* de realização de cada tarefa;
- Agendamento ou escalonamento: Determina *quando* será realizada cada tarefa (quando inicia e quando finaliza) satisfazendo todas as restrições exigidas no problema.

Por ser um problema importante na tomada de decisão, o problema de *scheduling* pode compreender diferentes critérios de desempenho do sistema, no qual pode ser escolhido para definir a melhor sequência de tarefas para o respectivo problema, respeitando suas restrições. Segundo Reklaitis (1982), alguns critérios de desempenho de sistema podem ser:

- *Makespan*: Tempo total para o processamento de todas as tarefas;
- *Mean flow-time*: Média de tempo que dura o fluxo de tarefas em todo o sistema;
- *Total flow-time*: Soma do tempo total de duração do fluxo de tarefas;
- *Mean tardiness*: Atraso máximo considerado para a conclusão de tarefas;

- *Tardiness*: Soma das penalidades devido ao atraso;
- *Earliness*: Soma das penalidades pelo adiantamento.

Segundo Hochbaum (1999) os problemas de sequenciamento têm os seguintes objetivos:

- Minimizar o tempo de execução total das tarefas, analisando os níveis de utilização das máquinas;
- Minimizar o tempo de espera de cada tarefa (o tempo entre o término de uma tarefa e o começo de outra);
- Minimizar os custos de execução de cada atividade.

De acordo com Loureiro (2014), o problema de *scheduling* tem forte aplicação na área de planejamento e controle da produção (PCP), e de modo geral tem como objetivo definir os tempos de início e de finalização que um conjunto de tarefas devem ser processadas por um conjunto de recursos, de forma a otimizar uma medida de desempenho relacionada ao tempo. Para Joo e Kim (2015), a etapa de *scheduling* é de grande impacto para o planejamento e controle da produção, por se tratar da programação de uma série de atividades que tornam disponíveis os recursos necessários para a conclusão das tarefas, determinando período de início para a produção.

Para Fernandes (2021), o modo em que a linha de produção está funcionando pode afetar a ordem dos processos, logo o problema de sequenciamento de tarefas está relacionado à forma em que está o posicionamento de máquinas. Portanto, a seguir será apresentado problema de sequenciamento em máquinas e seus respectivos modelos.

2.2. SEQUENCIAMENTO EM MÁQUINAS

De acordo com Santo (2014), o sequenciamento de operações, ou de tarefas, definirá em que ordem será feita a fabricação dos produtos, respeitando suas prioridades e restrições. Antes de realizar o sequenciamento deve-se todas as informações necessárias para a operação, como a capacidade de produção, o número

de máquinas disponíveis, o número de produtos com seus respectivos tempos de processamento e datas de entrega.

Segundo Pinedo (2008), o sequenciamento de tarefas em máquinas pode ser classificado segundo os atributos das máquinas utilizadas e o ambiente de trabalho. Serão apresentadas a seguir os possíveis ambientes de máquinas segundo este mesmo autor.

2.2.1. Modelo de máquina única

Neste modelo considera-se o recurso de apenas uma máquina trabalhando para realizar um conjunto de tarefas. É um problema simples que consiste na ordenação das operações na máquina. Para Loureiro (2014), é um problema de sequenciamento que tem como objetivo organizar uma sequência de tarefas de modo a minimizar um determinado parâmetro.

De acordo com Pinedo (2008), é um modelo mais simples comparando os demais modelos, em que nesse modelo todo o conjunto de tarefas é processado no mesmo recurso, podendo o recurso ser uma máquina, uma célula de produção ou um conjunto de recursos sendo considerado único. Na Figura 2 tem-se uma representação de processamento em máquina única.

Flizicoski (2017) realiza um estudo em um sistema de máquina única responsável por processar *n* tarefas previamente disponíveis, a máquina é representada por uma calandra que possuí a função de atribuir o formato laminado a polímeros, em que a máquina realiza a operação de três grupos de produtos, sendo eles: mantas de borracha, tecidos impregnados com borracha e aplicação de coberturas de borracha a materiais. Para resolver esse problema o autor envolve experimentos com modelagem matemática e heurísticas construtivas.

Figura 2 – Modelo de máquina única

Fonte: Santo (2014, p.19).

Na Figura 2 ilustra-se uma situação em que um único produto entra na máquina por vez, onde esse produto é processado por ela e após essa etapa, o produto é imediatamente liberado.

2.2.2. Modelo de máquina em paralelo

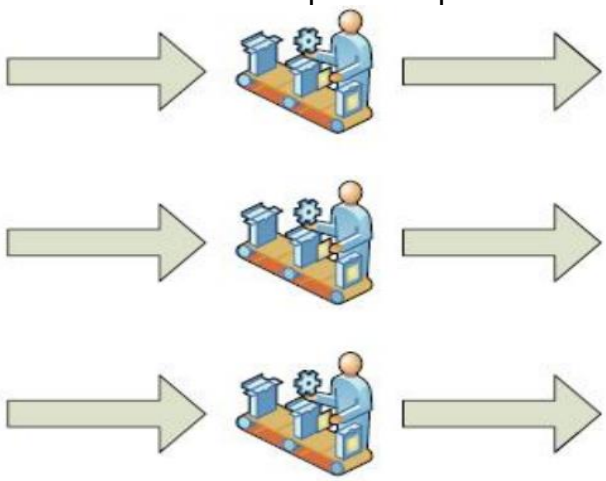
Neste modelo considera-se o uso de várias máquinas que realizam a mesma operação, podendo elas terem velocidades de processamento iguais ou diferentes de si. Segundo Reis, P. (2020), o planejamento de problemas para esse modelo possibilita a execução simultânea de um conjunto de tarefas, onde cada trabalho é constituído por uma operação. De acordo com Loureiro (2014), os problemas de máquinas paralelas podem ser divididos da seguinte maneira:

- Máquinas idênticas: Tempo das máquinas são iguais para processar uma tarefa;
- Máquinas uniformes: Cada máquina tem sua respectiva velocidade, então se uma máquina é duas vezes mais rápida que outra, logo ela processa a mesma tarefa na metade do tempo;
- Máquinas não relacionadas: Não há relação entre as velocidades de processamento das diferentes máquinas.

Fernandes (2021) informa que no caso de máquinas em paralelo com tempos de processamento diferentes, a velocidade de processamento de cada máquina interfere diretamente no tempo total de processamento das tarefas, visto que quanto antes uma máquina terminar uma tarefa mais cedo a máquina poderá ter uma nova alocação de tarefas.

Barbosa (2017) estuda um problema de sequenciamento de tarefas em 24 máquinas paralelas com tempos de processamento diferentes que executam 59 tarefas. Com o objetivo de reduzir o atraso de produção ao se concluir todas as tarefas ou ordens de produção. Para isso, a autora propõe um modelo de programação linear inteira mista e apresenta-se uma forma de resolução exata do modelo proposto através de *solvers.*

Figura 3 – Modelo de máquina em paralelo



Fonte: Santo (2014, p.20)

Na Figura 3 tem-se uma representação de processamento de máquinas em paralelo, em que diversos produtos entram por vez no ambiente de processo, onde esses produtos são direcionados e processados por várias máquinas em paralelo. Neste modelo podem ser processados inúmeros produtos simultaneamente.

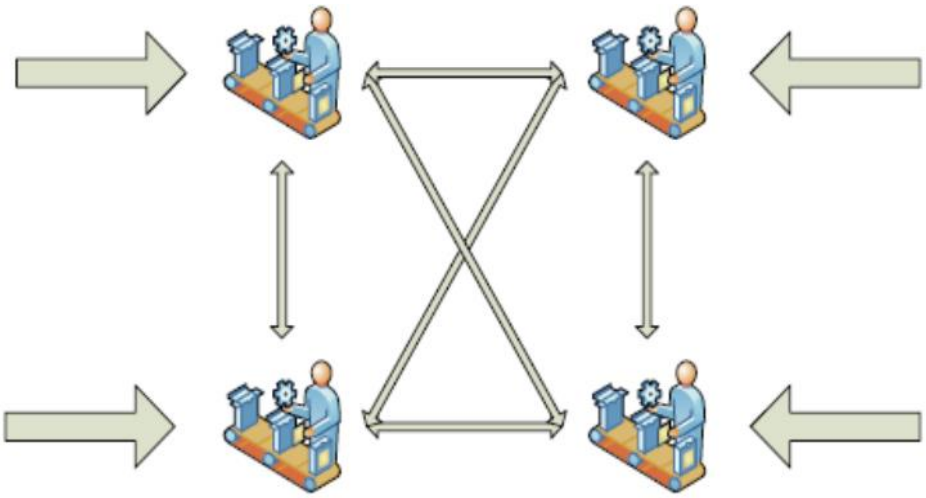
2.2.3. Modelo de *job shop*

Neste modelo considera-se o uso de várias máquinas, e cada máquina simboliza um centro de trabalho em que cada operação possui seu próprio roteiro, onde os produtos não são obrigados a passarem por todas as máquinas e nem seguir as mesmas sequências de produção. Segundo Barbosa (2017), este modelo consiste em um sistema de n peças e m máquinas, em que o processamento de cada peça consistem em x operações realizadas em uma sequência específica de um conjunto de tarefas.

De acordo com Veríssimo (2016), nesse modelo existem inúmeros conjuntos de tarefas que possuem um roteiro pré-determinado de máquinas, podendo visitar a mesma máquina várias vezes, tendo a ordem dessas operações específica das tarefas. Para Pinedo (2008), no modelo *job shop,* os roteiros pré-determinados de máquinas podem ser uma sequência de linhas de produção em que os produtos visitam uma determinada ordem de máquinas uma única vez, ou podem visitar uma máquina mais de uma vez, como pode ser visto na Figura 4.

Dias (2015) realiza um estudo de caso em uma indústria de cosméticos localizada na Região Metropolitana de Curitiba, em um ambiente *job shop* com *n* tarefas e *m* máquinas, na seção de acabamento, em que cada produto, denominado tarefa, pode seguir roteiros diferentes de acordo com sua embalagem. Com o objetivo de minimizar o *makespan*, é implementado um modelo exato no *software* Lingo.

Figura 4 - Modelo de *job shop*



Fonte: Santo (2014, p.21)

É possível observar na ilustração da Figura 4 que o produto pode entrar em qualquer máquina e seguir qualquer outro fluxo de produção, e até mesmo voltar a uma máquina já visitada.

2.2.4. Modelo de *flow shop*

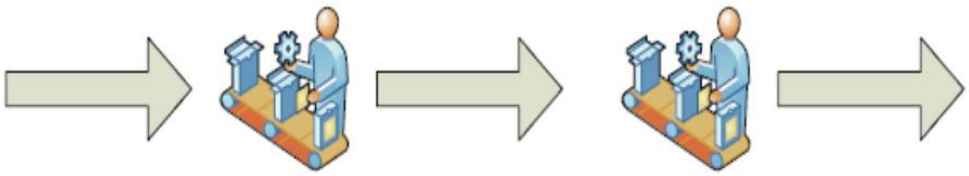
Neste modelo considera-se um esquema de fila semelhante ao *First In First Out*, que seria, o primeiro produto que entra no ambiente de produção é o primeiro a sair. Pinedo (2008) informa que o ambiente compreende *m* máquinas em série, em que cada produto deve ser processado por todas elas, em uma sequência pré-determinada antes de sair do ambiente.

Para Fernandes (2021), este modelo é um caso típico de *job shop*, pelo fato de todos os *jobs* possuírem a mesma sequência de produção com várias operações em série, em que cada operação deve ser processada em todas as máquinas de

acordo com a ordem já estabelecida antes do início do processo, como pode ser visto na Figura 5.

Vasquez (2017) realiza um estudo em um ambiente *flow shop* com *m* máquinas em série que devem processar *n* tarefas, com mais de duas máquinas, em que todas as tarefas têm uma data de entrega comum e restritiva, com o objetivo de minimizar a soma total dos adiantamentos e atrasos das tarefas em relação a data de entrega. Para solucionar o problema o autor apresenta uma solução heurística baseada em busca local, e para validar a abordagem é desenvolvida uma formulação matemática em programação linear inteira mista.

Figura 5 - Modelo de *flow shop*



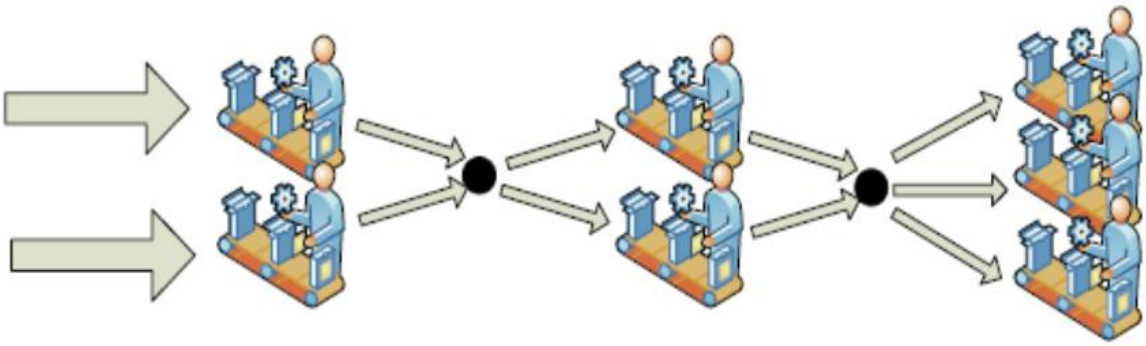
Fonte: Santo (2014, p.20)

É possível visualizar na Figura 5 que o produto entra no sistema e é processado pela primeira máquina, e logo em seguida, é processado pela segunda máquina, e após concluir sua sequência pré-estabelecida de produção, o produto sai do ambiente.

2.2.5. Modelo de *flexible flow* shop

Neste modelo considera-se um ambiente que possui várias máquinas em paralelo idênticas agrupadas em conjuntos, e esses conjuntos estão ligadas em série, no qual as filas dos produtos entre os conjuntos não precisam ser processadas por ordem de chegada (*First In First Out).* De acordo com Pinedo (2008), este modelo é uma generalização do modelo *Flow Shop* junto com um ambiente de máquinas em paralelo, porém ao contrário de existir *m* máquinas em série, este modelo considera *c* conjuntos em série e, em cada conjunto de máquinas, existem máquinas iguais em paralelos. Na Figura 6 a seguir ilustra-se um exemplo deste modelo.

Figura 6 - Modelo de *flexible flow shop*



Fonte: Santo (2014, p.21)

Na Figura 6 é possível observar o primeiro conjunto de máquinas em paralelo, a seguir, o símbolo representado pelo círculo preenchido em preto, ilustra o ponto intermediário entre os conjuntos de máquinas em paralelo.

O problema considerado neste trabalho considera um ambiente de sistema de manufatura flexível (FMS) composto por máquinas em paralelo, com sequenciamento de tarefas e dois robôs móveis. Após ter todas as informações e características do problema, deve-se focar no método de solução a ser proposto, na próxima seção será abordado este tópico.

2.3. ROBÔS AUTÔNOMOS E SEQUENCIAMENTO DE TAREFAS

A seção a seguir é dedicada a demonstrar o conceito de robôs autônomos, suas aplicações e tendências, e apresenta o problema estudado por Dang et al. (2013), com as adaptações propostas em (FERNANDES; TAGLIALENHA, 2021) e detalhar a modelagem matemática a ser utilizada para resolver o problema considerado neste trabalho.

2.3.1. Robôs autônomos

Robôs são agentes artificiais ativos, possuindo a capacidade de se mover no mundo real. São considerados artificiais por não serem entidades da natureza e ativos por desempenhar tarefas e se locomoverem no mundo físico real, de modo racional. (RIBEIRO; COSTA; ROMERO, 2001). A atuação do robô está ligada com a tarefa que irá desempenhar e ao ambiente no qual está inserido, podendo este ambiente ter

propriedades complexas, necessitando de requisitos particulares do projeto. (PACHECO; COSTA, 2002).

2.3.1.1 Aplicações

O estudo da robôs móveis é um tema bastante relevante, esta área de estudos, pesquisa e desenvolvimento tem apresentado um grande salto em seu desenvolvimento nas últimas duas décadas. A utilização de robôs móveis em diversas atividades da sociedade tem demonstrado que está área tem um grande futuro. (WOLF; OSÓRIO; TRINDADE JR., 2009). Segundo Andrade (2011), o robô móvel está sendo utilizado em vários ambientes, podendo ser industrial, doméstico ou militar. Essas aplicações podem ser em trabalhos insalubres ou perigosos ao ser humano, como em ambientes perigosos, na verificação de minas terrestres, entre outros.

Andrade (2011) cita alguns exemplos de aplicações de robôs móveis:

- Desastres nucleares, como o caso de Chernobil (1986) e Fukushima (2011), em que foram empregados os robôs móveis devido ao alto índice de radiação;
- Robôs submarinos que podem localizar e recolher destroços de acidentes;
- Robôs utilizados em entretenimento, como o futebol de robôs;
- Robôs em ambientes domésticos, como os utilizados para aspirar o pó;
- Robôs em ambientes de manufatura para executar tarefas, como o que será apresentado neste trabalho;

2.3.1.2 Tendências

Ribeiro (2022) traz um estudo feito pela empresa de pesquisas Alled Market Research, que informa que o mercado global de robôs móveis autônomos pode chegar a US$ 12,4 bilhões em 2030, podendo ter a uma taxa anual de crescimento de 17% entre 2021 e 2030, principalmente pela alta demanda oriunda de setores industriais, como o automotivo e de saúde. Segundo o autor a projeção para a utilização dos AMRs nos centros de armazenamento é de um aumento de 20% ao ano, devido a alta demanda de tarefas referentes a coleta e colocação de objetos em fábricas e instalações de fabricação.

A InforChannel (2022) publicou um levantamento feito pela empresa multinacional ABB, que realizou uma pesquisa com 250 empresas de vários setores e países, e identificou três tendências que moldarão a demanda por robôs móveis. A primeira é em relação a mudança global por veículos elétricos, esta alta demanda está impactando diretamente no ambiente fabril, está transição trará uma aceitação nos AMRs, pois permitirá que os fabricantes otimizem a entrega de componentes nas instalações. A segunda tendência é em relação ao aumento de procura do comércio eletrônico, para satisfazer esta alta demanda, os AMRs auxiliaram nos processos logísticos internos dos armazéns. A última tendência é em relação aos profissionais capacitados em utilizar os AMRs, tendo eles que ter conhecimentos de ferramentas de simulação e linguagens de programação avançadas, de modo a possuir habilidades de programar e operar os robôs.

A Mordor Intelligence (2022) traz algumas tendências para o mercado de veículos não tripulantes (AGV - *Unmanned Ground Vehicles*). O primeiro é em relação a executar tarefas como segurança de perímetros de áreas perigosas, realização de operações de resgate em desastres e manuseio de materiais perigosos. Outra tendência é em relação ao crescente uso de câmeras de imagem térmica para garantir a navegação autônoma do robô, utilizado principalmente em ameaças terroristas. A utilização de sensores LiDAR também é outra tendência, visto que fornecem uma navegação precisa.

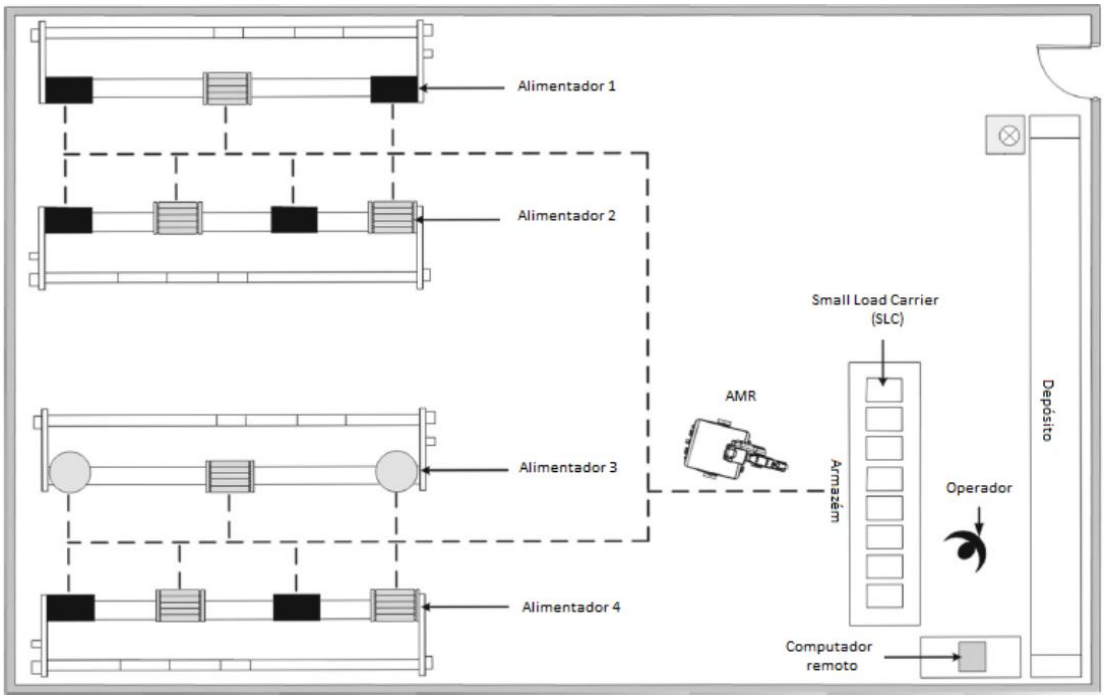
2.3.2. Definição do problema de *scheduling* considerando um robô

O problema estudado no trabalho de Dang et al. (2013) considera robôs móveis autônomos (AMR) com braços mecânicos que realizam tarefas referentes a transportar, coletar e alimentar máquinas com pequenas caixas padronizadas de peças (SLC – *Small Loaded Carries*), em uma linha de produção estruturada considerando-se máquinas em paralelo. Para a realização dessas tarefas (alimentação de peças) neste estudo considera-se um AMR, o qual, deve ser programado de modo a realizar a alimentação e evitar a falta de peças na linha de produção, e respeitando ainda, as restrições de capacidade de estocagem de peças nas máquinas, capacidade de carregamento e autonomia de bateria dos robôs.

Na Figura 7 ilustra-se o ambiente da linha de produção, considerado em Dang et al. (2013), o qual consiste em quatro alimentadores de máquinas que devem ser

abastecidos pelo AMR e um armazém. Vale ressaltar que máquinas diferentes devem ser alimentadas por diferentes tipos de peças.

Figura 7 - Layout do ambiente de produção



Fonte: Dang et al. (2013, p.1273).

Na Figura 7 ilustra-se uma área de uma fábrica dividida em células de manufatura com duas linhas de produção, a primeira entre os alimentadores 1 e 2, e a outra entre os alimentadores 3 e 4. Os AMRs têm a responsabilidade de realizar as tarefas de posicionamento das caixas ao lado das máquinas e de abastecer os alimentadores das máquinas com suas respectivas peças. Para realizar essas tarefas, Dang et al. (2013), adotou o conceito de *bartender*.

No conceito de *bartender*, cada alimentador possuirá os seguintes atributos, nível máximo de peças, nível mínimo de peças e uma taxa de alimentação de peças na máquina. Além disso, os *bartender*s são responsáveis por colocar as peças em pequenas caixas padronizadas (SLC – *Small Loaded Carries*), que ficam estocadas no armazém central (bar). Durante o procedimento, o robô pega uma quantidade informada de SLCs do armazém central e vai até a máquina que foi designado e descarrega as peças no alimentador, e viaja até o bar para descarregar os SLCs

vazios e carregar novamente, e assim recomeça a operação até não ter mais tarefas a serem executadas.

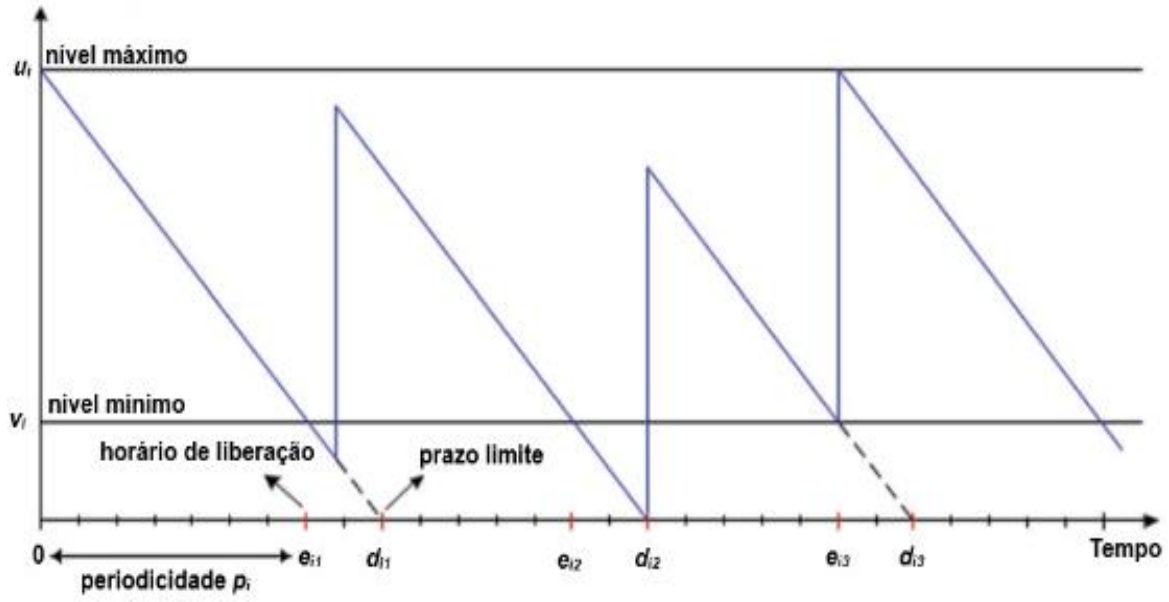
Quando se considera a realização das tarefas por um único robô, tem-se o SAMR (*Scheduling a single AMR*), segundo (DANG et al., 2013).

2.3.3. Modelagem matemática para o SAMR

Dang et al. (2013) traz a modelagem matemática para considerar um robô, no qual a proposta do autor tem como premissas que são conhecidas todos os dados das tarefas (alimentação de peças nos alimentadores) antecipadamente, bem como a capacidade de carregamento do AMR.

Para garantir as restrições de capacidade de estocagem das máquinas e evitar a falta de peças durante o *lead time* de reposição, definiram-se restrições de janela de tempo para a realização das tarefas de alimentação baseado no sistema de inventário de estoque (s, Q), ilustrado na Figura 8, o qual leva em consideração os atributos de tempo de processamento das máquinas.

Figura 8 - Representação da janela de tempo



Fonte: Baseado em Dang et al. (2013).

Assim, nesse método, no instante em que o nível de peças da máquina chega a um nível s, ilustrado na Figura 8 como v_i , é realizado um pedido de reposição Q. Além do mais, foi considerada também a capacidade de carga, a velocidade do robô e a taxa de alimentação de peças nos alimentadores. Com base nessas informações, é realizado o cálculo da periodicidade, informando em que período deve ser realizado a reposição de peças com base no nível do estoque, ou seja, quanto mais perto do valor da periodicidade menor o nível de estoque.

O objetivo é minimizar o tempo total de viagens de um AMR para um determinado horizonte de planejamento, considerando-se as seguintes premissas:

- O AMR é posicionado inicialmente no armazém;
- As tarefas são periódicas, independentes e atribuídas ao mesmo AMR (SAMR);
- A taxa de alimentação de peças do alimentador para a máquina é conhecida e constante;
- O tempo de trabalho e de viagem do AMR entre qualquer par de localidades são conhecidas;
- Todos os alimentadores devem ser carregados até o volume máximo de peças suportado.

Segundo Fernandes (2021), conforme aumenta a complexidade do problema, o tempo demandado para obter a solução aumenta exponencialmente. Com isso, na prática o modelo de PLIM (Programação Linear Inteira Mista) apenas poderá ser aplicado em menor escala, com uma combinação dos poucos alimentadores na linha de produção, número de subtarefas baixo e um horizonte de planejamento curto.

Considerando as premissas anteriores, o modelo PLIM irá resultar em uma solução ótima, o que é utilizado como parâmetro comparativo com o modelo heurístico implementado por Dang et al. (2013).

Os parâmetros de entrada do modelo assumem as seguintes notações:

- N: grupo de todas as tarefas ($N = 0, 1, 2, \ldots, n,$ onde 0 é a tarefa no armazém);
- n_i : número de vezes que a tarefa i tem que ser executada;

- R: grupo de todas as possíveis rotas ($R = 1, 2, \ldots, Rmax$; $Rmax = \sum n_i \ , \forall i \in N \setminus \{0\}$);
- e_{ik}: horário de liberação da subtarefa k da tarefa i em segundos;
- d_{ik}: prazo limite da subtarefa k da tarefa i em segundos;
- p_i : tempo de periodicidade da tarefa i em segundos;
- w_i : tempo de processamento do AMR na localização i em segundos;
- t_{ij} : tempo de viagem do AMR da localização i até a localização j em segundos;
- c_i : taxa de alimentação de peça do alimentador i para máquina em segundos;
- v_i : volume mínimo de unidades peças no alimentador i;
- u_i : volume máximo de unidades peças no alimentador i;
- Q_i : número máximo de SLC que o AMR consegue carregar;
- T: horizonte de planejamento em segundos;

As variáveis de decisão usadas no modelo são:

- X_{ik}^{jlr} = 1, se o AMR viaja da localização da tarefa i, subtarefa k, para a localização da tarefa j, subtarefa l, na rota r; 0, caso contrário;
- Y_{ik} :índice da rota cuja a subtarefa k da tarefa i pertence;
- S_{ik}:tempo de início da subtarefa k da tarefa i.

Devido às tarefas terem características periódicas nos alimentadores, é necessário calcular sua periodicidade, a qual é dada pela Equação (1), resultando em um número de subtarefas, o número de subtarefa é definido pela equação $n_i = [T/p_i]$.

$$p_i = (u_i - v_i) * c_i, \forall i \in N \setminus \{0\} \quad (1)$$

O robô deve iniciar o processamento de uma subtarefa k da tarefa i dentro da janela de tempo $[e_{ik}, d_{ik}]$ associada a essa subtarefa. Portanto, o robô não pode processar a tarefa do alimentador i após o limite superior da janela de tempo. Caso o

robô chegue no alimentador i antes do horário inferior da janela de tempo e_{ik}, o robô terá que esperar para iniciar o serviço.

O limite inferior da janela de tempo, ou o horário de liberação, é definido como o tempo em que o número de peças dentro do alimentador atinge o nível v_i ,, e é dado pela Equação (2).

$$e_{ik} = e_{(ik-1)} + p_i \ , \forall i \ \in \ N\backslash\{0\}, k \in 1,2, \dots, n_i \ , e_{i0} = 0 \tag{2}$$

Já o limite superior da janela de tempo da subtarefa k da tarefa i, é definido para o momento em que não há peças no alimentador, e é representado pela Equação (3).

$$d_{ik} = e_{ik} + \left(v_i \ * c_i \ \right), \forall i \ \in \ N\backslash\{0\}, k = 1,2, \dots, n_i \tag{3}$$

A função objetivo definida na Equação (4) minimiza o tempo total dos deslocamentos dos robôs entre as máquinas. Destaca-se que neste modelo, o tempo de trabalho do robô em cada máquina e a quantidade de operações nos alimentadores em cada máquina são conhecidos previamente.

$$Min\ Z = \sum_{i \in N} \sum_{k=1}^{n_i} \sum_{j \in N} \sum_{l=1}^{n_j} \sum_{r \in R} t_{ij} * X_{ik}^{jlr} \tag{4}$$

Os conjuntos de restrições utilizados no problema são:

$$e_{ik} < S_{ik} < d_{ik}, \forall i \ \in \ N\backslash\{0\}, k \in 1,2, \dots, n_i \tag{5}$$

$$\sum_{j \in N\{0\}} \sum_{l=1}^{n_j} x_{01}^{jl1} = 1 \tag{6}$$

$$\sum_{j \in N\backslash\{0\}} \sum_{l=1}^{n_j} \sum_{r \in R} X_{01}^{jlr} \leq 1 \tag{7}$$

O conjunto de Restrições (5) garantem que o tempo de início de processamento da subtarefa k da tarefa i, respeite os limites inferior e superior da janela de tempo. As Restrições (6) farão com que o robô inicie o problema no armazém, enquanto que o conjunto de Restrições (7) fará com que esse estágio inicial da Restrições (6) aconteça apenas uma vez.

O conjunto de Restrições (8) garante a eliminação das sub-rotas entre as subtarefas de tarefas, em que Z é um subconjunto de Z_T, sendo que Z_T é o conjunto de todas as subtarefas de tarefas nos alimentadores e no armazém.

$$\sum_{(i,k),(j,l)Z} X_{ik}^{jlr} \leq |Z| - 1, \forall r \in R, \forall Z \in Z_T, Z_T = \{(i,k)|i \in N\{0\}, k \in 1,2,\dots,n_i\ \} \quad (8)$$

O conjunto de Restrições (9) e (10) garantem que apenas uma subtarefa de uma tarefa será realizada imediatamente após e imediatamente antes da subtarefa k da tarefa i, respectivamente.

$$\sum_{j \in N} \sum_{l=1}^{n_j} \sum_{r \in R} X_{ik}^{jlr} = 1, \forall i\ \in\ N\backslash\{0\}, k \in 1,2,\dots,n_i \quad (9)$$

$$\sum_{i \in N} \sum_{k=1}^{n_i} \sum_{r \in R} X_{ik}^{jlr} = 1, \forall j\ \in\ N\backslash\{0\}, k \in 1,2,\dots,n_j \quad (10)$$

As Restrições (11) fazem com que o robô não carregue mais SLC que sua capacidade permitida.

$$\sum_{i \in N} \sum_{k=1}^{n_i} \sum_{j \in N\backslash\{0\}} \sum_{l=1}^{n_j} X_{ik}^{jlr} \leq\ Q_m, \forall r\ \in\ R \quad (11)$$

O conjunto de Restrições (12) garante que os tempos de viagem entre localidades sejam garantidos, essa mesma restrição assegura que o tempo de início da subtarefa k da tarefa i não seja maior que o da subtarefa i da tarefa j, caso elas aconteçam em sequência. L é uma constante muito grande.

$$s_{ik} + (w_i\ + t_{ij} \sum_{r \in R} X_{ik}^{jlr}) - L(1 - \sum_{r \in R} X_{ik}^{jlr}) + (y_{jl}\ - y_{ik})(t_{i0} + w_0\ t_{0j} - t_{ij}\ ) \leq s_{jl} \quad (12)$$
$$\forall i,j\ \in\ N, k \in 1,2,\dots,n_i\ , l \in 1,2,\dots,n_j$$

As Restrições (13) atribuem uma subtarefa de uma tarefa a uma rota r.

$$y_{jl} = \sum_{i \in N} \sum_{k=1}^{n_i} \sum_{r \in R} r * X_{ik}^{jlr}, \forall j \in N \backslash \{0\}, l \in 1,2,\dots,n_j \quad (13)$$

O conjunto de Restrições (14) garante que os índices das rotas utilizadas aumentem sequencialmente.

$$y_{jl} \geq y_{ik} \sum_{r \in R} X_{ik}^{jlr}, \forall i,j \in N, k \in 1,2,\dots,n_i, l \in 1,2,\dots,n_j \quad (14)$$

E por fim, as Restrições (15) e (16) indicam os domínios das variáveis.

$$X_{ik}^{jlr} \in \{0,1\}, \forall r \in R, \forall i,j \in N, k \in 1,2,\dots,n_i, l \in 1,2,\dots,n_j \quad (15)$$

y_{ik}: inteira positiva $\forall i \in N \backslash \{0\}, k \in 1,2,\dots,n_i$ (16)

S_{ik}: não negativa $\forall i \in N \backslash \{0\}, k \in 1,2,\dots,n_i$ (17)

2.3.4. Modelagem matemática para considerar vários AMRs

Para possibilitar a utilização de mais de um robô na execução das tarefas, além das premissas citadas na seção anterior, foi necessário acrescentar mais alguns parâmetros para adaptar o modelo, sendo eles:

- Nv: Número de robôs;
- N: Grupo de todas as tarefas;

Além disso, visando minimizar o tempo total para processar todas as tarefas pelos robôs, neste trabalho considerou-se o modelo PLIM dado pelas Equações (1)-(17), entretanto, a função objetivo dada na Equação (4) foi alterada para, além de considerar o tempo de viagem dos robôs entre as máquinas, considerar também o tempo de trabalho e o tempo de retorno para o armazém de cada robô. Assim, definiram-se a variável de decisão:

- $X_{ikv}^{jlr} = 1$, se o robô v viaja da localização da tarefa i, da subtarefa k, para a localização da tarefa j, dá subtarefa l, na rota r; e $X_{ikv}^{jlr} = 0$, caso contrário.

A função objetivo considerada neste trabalho é dada pela Equação (18):

$$Min\ Z = \sum_{i=1} w_i + \sum_{i \in N} \sum_{k=1}^{n_i} \sum_{j \in N} \sum_{l=1}^{n_j} (t_{ij} + (y_{ij} - y_{ik})(t_{il} + t_{lj} - t_{ij})) * \sum_{r \in R} \sum_{v=1}^{Nv} X_{ikv}^{jlr} \quad (18)$$

A primeira parcela da Equação (18) representa o *makespan* ao ser somado com a segunda parcela, resultando no tempo total para processar todas as tarefas do estudo em questão.

Considerando situações práticas em que o tempo de processamento dos robôs é diferente, define-se uma nova FO dada pela Equação (19), considerando-se um novo parâmetro w_i^v:

- w_i^v: Tempo de processo do robô v na localização i.

$$Min\ Z = \sum_{i \in N} \sum_{k=1}^{n_i} \sum_{j \in N} \sum_{l=1}^{n_j} \sum_{v=1}^{Nv} w_i^v * X_{ikv}^{jlr} + \sum_{i \in N} \sum_{k=1}^{n_i} \sum_{j \in N} \sum_{l=1}^{n_j} (t_{ij} + (y_{ij} - y_{ik})(t_{il} + t_{lj} - t_{ij})) * \sum_{r \in R} \sum_{v=1}^{Nv} X_{ikv}^{jlr} \quad (19)$$

2.3.5. Complexidade do problema de otimização

Na área da pesquisa operacional, os modelos exatos fornecem soluções ótimas para os problemas de otimização, porém não são todos os problemas que podem utilizar essa modelagem de solução. Determinados problemas são complexos de modo que pode não ser possível encontrar uma solução ótima em tempo polinomial em função da entrada de dados, entretanto ainda é necessário encontrar uma solução viável de boa qualidade, de modo que seja próxima da solução ótima (HILLIER; LIEBERMAN, 2013).

Existem diferentes níveis de complexidades para os problemas de otimização, podendo a complexidade ser medida pelo tempo computacional necessário para obter a solução, pela quantidade de memória utilizada, ou qualquer outra medida de desempenho que analise a utilização de um algoritmo. Comumente a medida utilizada para medir a qualidade de um algoritmo é o tempo computacional (COLIN, 2007).

Segundo Filho (2008), com a utilização da Teoria da Complexidade é possível classificar a classe em que o problema de otimização combinatória se encontra de forma eficiente, podendo determinar o grau de dificuldade da complexidade para resolver determinado problema. Segundo essa teoria os problemas podem ser classificados da seguinte maneira:

- Problemas da Classe P (*Polinomial Time*): São problemas que possuem solução em tempo polinomial de acordo com o tamanho da entrada de dados. São considerados de fácil solução, implementação e mais eficiente (COLIN, 2007).
- Problemas da Classe NP (*NonDeterministic Polinomial Time*): São problemas em que seus algoritmos de solução são baseados em enumeração. De modo geral, possui grande quantidade de combinações, fazendo com que seu algoritmo não consiga resolver problemas de grande porte em tempo hábil. São denominados algoritmos de tempo exponencial (COLIN, 2007).
- Problemas NP - Completos (*completed*): São os mais difíceis da classe NP, são problemas pertencentes à interseção das classes NP e NP-Difícil (COLIN, 2007).
- Problemas NP - Difícil (*Hard*): Os problemas de otimização cujos problemas de decisão correspondentes são NP - completos são denominados NP - difíceis (COLIN, 2007).

Devido à sua natureza combinatorial, o problema do AMR é considerado um problema da classe *NP-Hard* (DANG et al., 2013).

2.4. MÉTODOS DE SOLUÇÃO PARA O PROBLEMA AMR

Dependendo de suas características, os problemas de programação linear podem trazer uma alta complexidade de resolução, e com isso, podem ser resolvidos considerando algumas abordagens, as quais podem trazer soluções exatas ou aproximadas.

Segundo Fernandes (2021), pelo fato de considerar as variáveis de decisão inteiras no modelo, isso resulta em um problema de otimização muito mais complexo de ser resolvido.

2.4.1. Solução via Modelos de programação matemática

Os métodos exatos encontram as soluções ótimas do problema, e para isso se faz necessário o uso de *softwares* específicos de otimização. Com os avanços tecnológicos foi possível o desenvolvimento de computadores com alta capacidade de processamento, tornando viável a resolução de problemas com número considerável de variáveis, como o complemento *Solver* do Excel.

De acordo com Fernandes (2021), o problema de AMR só pode ser resolvido com o método exato para situações de pequeno porte, com poucas máquinas, poucos robôs e poucas tarefas.

Com a utilização de linguagens específicas de modelagem matemática, como o AMPL (2018), é possível utilizar recursos computacionais, como o LINGO, GUROBI e CEPLEX, como ferramentas de solução de problemas gerais de otimização (ANAND; AGGARWAL; KUMAR, 2017).

Dang et al. (2013) propõe um modelo matemático para resolver o problema de sequenciamento de tarefas de alimentação de um único robô móvel autônomo em ambiente de manufatura flexível para diversos alimentadores. Um método baseado nas características dos alimentadores e inspirado no sistema de inventário (s,Q) é então aplicado para definir janelas de tempo para as tarefas de alimentação do robô. Para isso, um modelo de Programação Linear Inteira Mista (PLIM) foi desenvolvido para minimizar o tempo total de viagem do robô para um determinado horizonte de planejamento. O modelo foi aplicado em um estudo de caso em uma linha de produção de rotores.

Fernandes (2021) estudou o caso de Dang et al. (2013) e desenvolveu um modelo de um robô móvel autônomo, no qual busca minimizar o *makespan* - tempo total requerido para a execução de todas as tarefas - em uma linha de produção para

um horizonte de planejamento pré-estabelecido. A diferença de seu estudo com o de Dang et al. (2013) está na relação a formação da janela de tempo restrita, que continua sendo restrita, isto é, o processamento da tarefa deve ser feito obrigatoriamente dentro do intervalo de tempo, porém, o intervalo de tempo pode variar com o momento em que a tarefa anterior é realizada. Um método de solução exata que utiliza o modelo matemático de PLIM é implementada, em linguagem AMPL, e comparada com trabalhos anteriores.

Kim, Cho e Kwon (2021) apresentaram um modelo matemático para resolver a situação de *deadlock*, ou seja, colisão entre robôs, e reduzir o tempo de tarefa em um cenário onde vários robôs móveis autônomos trabalham simultaneamente e em um determinado ambiente de armazém logístico. Cada robô recebe inicialmente uma tarefa aleatória que se move do nó inicial para o nó de chegada e, quando ocorre um *deadlock* por outro robô, é necessário encontrar o caminho de desvio ideal para o objetivo. O ambiente de estudo é composto por quatro robôs móveis em um armazém logístico virtual e um espaço de (10m x 10m). Para resolver esse problema, um modelo PLIM foi desenvolvido para resolver o determinado problema, gerando o caminho ótimo para cada robô na situação de *deadlock*.

Riazi e Lennartson (2021) desenvolveram um modelo matemático para solucionar o problema de roteirização e programação de AGVs sem conflitos. O modelo proposto é dividido em um problema principal, que consiste em lidar com a atribuição e sequenciamento de tarefas, e um subproblema, que visa verificar a viabilidade da solução do problema principal. Por ser um problema complexo, os autores consideraram instâncias pequenas variando de 8 a 12 tarefas e de 4 a 12 AGVs. No trabalho foi utilizado dois *solvers* distintos, um de código aberto e outro comercial e uma heurística. De acordo com os resultados, ambos os *solvers* podem resolver os modelos propostos e a heurística alcança soluções de boa qualidade rapidamente.

Lager et al. (2022) propõe um modelo que considera um robô móvel autônomo que deve completar as tarefas referentes a transportar caixas de *kits* vazias e buscar um objeto de uma caixa de *kit* transportada em um ambiente que pode ter imprevistos no caminho e deve gerar uma solução em tempo real. Propõe um modelo baseado em grafos do aplicativo do robô é convertido em um problema de agendamento de tarefas a ser resolvido por uma abordagem proposta de *Branch and Bound* (B&B) e

duas abordagens de benchmark: Programação Linear Inteira Mista e Linguagem de Definição de Domínio de Planejamento.

2.4.2. Solução via Métodos heurísticos

Conforme o tamanho do problema aumenta, o tempo demandado para obter a solução do PLI aumenta exponencialmente. Com isso, na prática, a aplicação do PLIM apenas pode ser aplicada em menor escala, como dito anteriormente.

Quando nos deparamos com problemas que não podem ser resolvidos otimamente, procuramos desenvolver procedimentos “inteligentes” que encontram soluções tão boas quanto possíveis.

Para Hillier e Lieberman (2013), o método heurístico é um procedimento inteligente que determina uma solução viável excelente para um problema de difícil solução, porém não necessariamente com a garantia de que a solução ótima do problema seja encontrada.

As heurísticas são conhecidas como métodos aproximados devido a serem capazes de resolver problemas de grande porte para um limite de tempo. E quanto melhor for a heurística, mais próximo da solução ótima ela chega.

As heurísticas podem ser classificadas em três categorias:

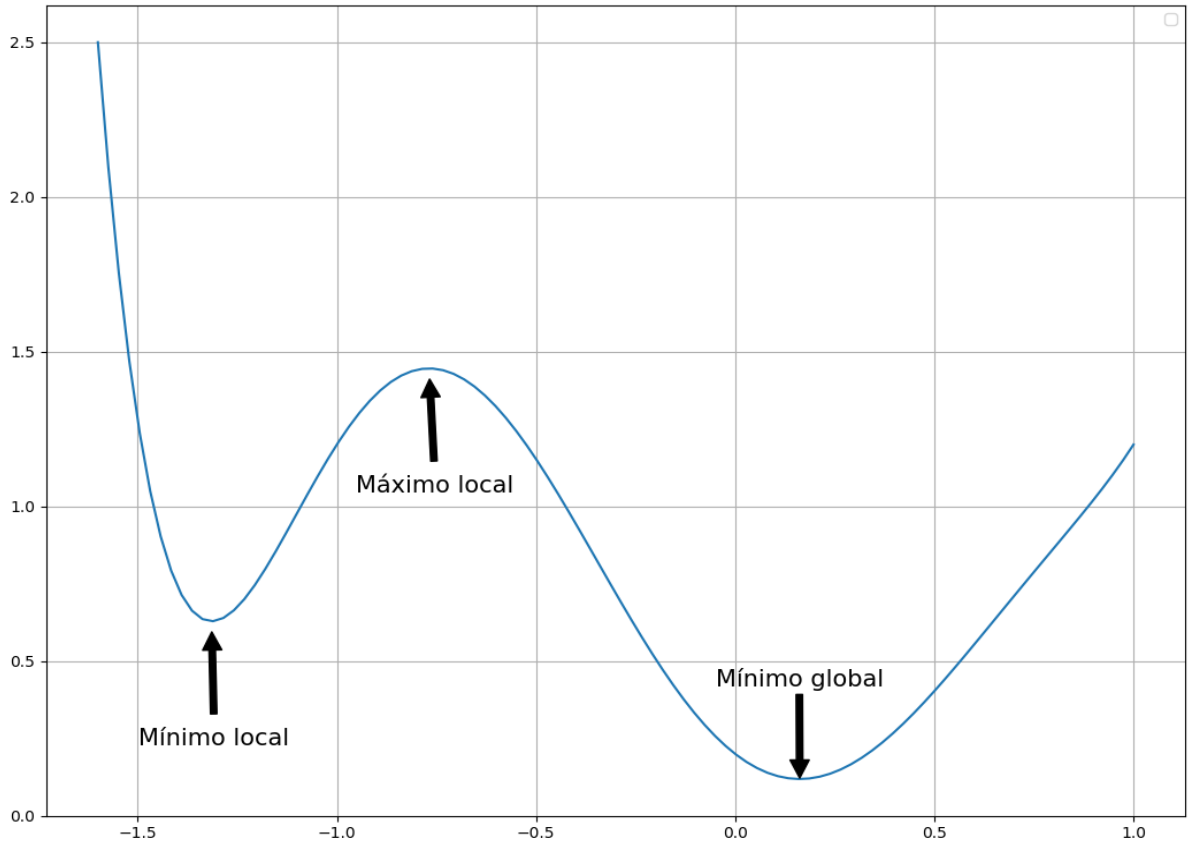
- Construtivas: Consiste em gerar uma solução passo a passo respeitando as restrições do problema. Segundo Cui et al. (2019), uma heurística construtiva constrói uma solução que se inicia vazia e expande a lista de tarefas da solução até processar todas as tarefas consideradas no problema, segundo algum critério de desempenho para definir a escolha da próxima tarefa a ser processada;
- Melhoria: Conhecida também como busca local, são heurísticas que utilizam como base uma solução viável inicial, a qual são realizadas manipulações a fim de obter uma melhor solução. Para Santos (2016), dada uma solução inicial viável, percorre-se a vizinhança da solução atual em busca de uma solução de melhor qualidade segundo uma medida de desempenho a ser otimizada. Em geral, a busca converge para uma solução ótima local;

- Híbridas ou composta: É a combinação das categorias citadas anteriormente com objetivo de melhorar a solução de trabalho.

Um dos problemas relacionados à aplicação de métodos heurísticos em problemas multimodais, ou seja, problemas com muitas soluções ótimos locais, a forma de como escapar de ótimos locais é essencial para encontrar uma solução de melhor qualidade. Na Figura 9 ilustra-se uma função com multimodais e indicam-se soluções de mínimo locais e a solução ótima global.

Na Figura 9 é possível observar a existência de um máximo local, mínimo local e o mínimo global. Assumindo que a FO seja de minimização, a solução ótima do problema seria no ponto do mínimo global, visto que é o ponto onde a curva tem o menor valor.

Figura 9 - Função com ótimo global e local



Fonte: Autor (2022).

2.4.3. Solução via Métodos meta-heurísticos

A heurística é pensada para um problema específico. Já uma meta-heurística, pode ser aplicada para qualquer problema. A utilização de meta-heurísticas é justificada em problemas de alta complexidade, com grande número de variáveis e em casos em que há pouco tempo disponível para resolver determinado problema. Em geral, sua aplicação não garante que seja alcançada a solução ótima global do problema. Mesmo não garantindo a solução ótima, a utilização de métodos heurísticos e meta-heurísticos encontra soluções bem aproximadas para o problema, de uma forma simples, rápida e com tempo computacional aceitável (VARELA, 2007).

Segundo Santos (2016), as meta-heurísticas também podem ser classificadas da seguinte maneira:

- Meta-heurística baseada em população: Manipula um conjunto de soluções, denominada população, em que cada solução representa um indivíduo, permitindo uma maior diversificação no espaço de busca. Exemplos são: Algoritmo Genético, Algoritmos Evolucionários, Enxame de Partículas, *Scatter Search*, entre outros.
- Meta-heurística baseada em solução única: Utiliza apenas uma solução durante a busca, possibilitando intensificar a busca em regiões locais. Exemplos: *Simmulated Anealing, Variable Neighborhood Seach* e *Iterated Greedy*.

Dang et al. (2013) apresenta uma heurística baseada em algoritmo genético que resulta em um aumento significativo na velocidade de encontrar soluções quase ótimas para problemas de grande porte para resolver o problema de sequenciamento de tarefas de alimentação de um único robô móvel autônomo em ambiente de manufatura flexível para diversos alimentadores. Um método baseado nas características dos alimentadores e inspirado no sistema de inventário (*s,Q*) é então aplicado para definir janelas de tempo para as tarefas de alimentação do robô. Com objetivo de minimizar o tempo total de viagem do robô para um determinado horizonte de planejamento. A heurística foi utilizada em um estudo de caso em uma linha de produção de rotores, e seu resultado foi compara com o modelo de programação inteira mista.

Fernandes (2021) estudou o caso de Dang et al. (2013) e desenvolveu uma meta-heurística baseada no *Simulated Annealing*, no qual busca minimizar o *makespan* - tempo total requerido para a execução de todas as tarefas - em uma linha de produção para um horizonte de planejamento pré-estabelecido. Comparou-se os resultados obtidos com os de Dang et al. (2013) nos mesmos cenários.

O estudo apresentado por Poudel, Zhou e Sha (2021) mostra dois métodos de agendamento e estudos de casos em um ambiente de mudança dinâmica sem colisões, em que os robôs móveis disponíveis devem realizar as tarefas para fazer a impressão de determinadas peças. O ambiente é em manufatura aditiva, o que diferente da manufatura tradicional por permitir a impressão paralela, reduzindo o makespan. São consideradas restrições de tempo incertas, em que não se sabe quando ou onde a próxima tarefa de impressão ocorrerá antes do tempo, restrições espaciais, onde vários robôs móveis trabalham em conjunto e precisam realizar a impressão sem colidir uns com os outros, ou com a peça impressa, e também por considerar restrições lógicas, que representam a dependência entre as tarefas. Os autores trazem dois métodos de solução, o primeiro método, algoritmo de lista de dependência dinâmica (DDLA), usa uma abordagem de satisfação de restrições para eliminar soluções que podem resultar em colisões entre robôs e colisões entre robôs com materiais já impressos, o segundo método, algoritmo genético modificado (GA), usa cromossomos para representar atribuições de pedaços e utiliza operadores GA, como o cruzamento e mutação, para gerar diversos agendamentos de impressão, mantendo as dependências entre os pedaços. É resolvido três estudos de casos para apresentar a eficácia e o desempenho dos dois métodos.

Jun, Lee e Yih (2021) abordaram o problema de coleta e entrega considerando as características dos AMRs em ambientes de fabricação. Para resolver o problema, primeiro os autores propõem uma nova formulação matemática considerando estratégias de recarga parcial e total para minimizar o atraso total das solicitações de transporte. Cada AMR tem uma carga útil máxima, tempos de viagem inicial e máximo e uma velocidade, e sua bateria diminui linearmente com a distância. Posteriormente apresentam dois algoritmos heurísticos construtivos com alta velocidade de computação, que são chamados de Algoritmo de Agrupamento Iniciado por Solicitação de Transporte (TRIGA), que na solicitação de transporte primeiro seleciona um veículo e depois escolhe outras solicitações de transporte, sacrificando seu atraso em um determinado nível, e Algoritmo de Agrupamento Iniciado por Veículo (VIGA),

que possuí a ideia básica de que um veículo ocioso selecione uma ou mais solicitações de transporte de uma lista de solicitações de acordo com decisões de agrupamento. A seguir, desenvolvem um algoritmo memético (MA) que incorpora um algoritmo genético em técnicas de busca local para encontrar soluções quase ótimas em um tempo razoável, evitando convergência prematuras. E por fim, avaliam o desempenho dos algoritmos propostos em comparação com duas regras de despacho, sendo menor distância e tempo de vencimento.

Manafi, Tavakkoli-Moghaddam e Mahmoodjanloo (2022) estudaram o problema de sincronização de agendamento de tarefas e roteamento de AGV em um ambiente de manufatura. O objetivo do trabalho é minimizar o tempo total de conclusão de todas as tarefas (ou seja, *makespan*). Os autores ilustram um modelo PLIM para resolver o problema e propõem um algoritmo de otimização de recifes de coral baseado em oposição de centroides (COCRO) combinado com uma heurística para resolver problemas de grande escala. A heurística desenvolvida utiliza o algoritmo de *Dijkstra* para extrair caminhos sem colisões. Para avaliar os resultados do algoritmo proposto, um experimento computacional é projetado com base em vários problemas de instâncias aleatórias.

Em Vieira e Taglialenha (2022) apresenta-se um método para resolução de um problema de *scheduling* de dois robôs moveis autônomos em um ambiente de manufatura flexível composta por um armazém (*warehouse*) e quatro máquinas e seus respectivos alimentadores (*feeders*) que devem ser servidos pelos robôs móveis autônomos (AMRs). Para resolver o problema os autores propõem uma heurística construtiva controlada para obter uma solução inicial de boa qualidade, e uma heurística de vizinhança variável, no qual, aplicam-se buscas de descidas sistemáticas no entorno de vizinhanças da solução escolhidas de forma sistemática. Os autores consideram três estruturas de vizinhança, sendo elas, a *Internal insertion, internal swap* e *external insertion*.

Neste trabalho, será aplicado dois métodos de solução baseados em vizinhança variável, melhor explicada nas próximas seções.

2.5. *VARIABLE NEIGHBORHOOD SEARCH* - VNS

A meta-heurística *Variable Neighborhood Search* (VNS) foi proposta no final da década de 90 por Nenad Mladenovic e Pierre Hansen, e utiliza a ideia de explorar

o espaço de soluções através de trocas sistemáticas de estruturas de vizinhança durante o processo de busca (MLADENOVIC; HANSEN, 1997).

Segundo Mladenovic e Hansen (1999), uma estrutura de vizinhança no espaço de soluções S é uma aplicação $N: \rightarrow S = \{N_k(x): N_k(x) \subseteq X$, que a cada solução $x \in X$ associa um conjunto de soluções $N_k(x) \subseteq X$, denominado vizinhança de x.

Denotando por $N_k(k = 1, \ldots, kmax)$, um conjunto finito de estruturas de vizinhança pré-selecionadas e por $N_k(x)$ o conjunto de soluções na $x - ésima$ de x, em que as vizinhanças N_k podem ser caracterizadas por uma ou mais métricas (ou quase-métricas) introduzidas num espaço de solução X. Em geral, uma busca local de descida troca a solução atual por outra solução melhor na vizinhança considerada, portanto apresenta o risco de estacionar num mínimo local.

Ou seja, a meta-heurística de busca em vizinhança variável é uma extensão de um algoritmo de busca local que utiliza a estratégia de mudança de tamanho da vizinhança para sair de soluções ótimas locais, seguindo os passos do Algoritmo 1 e fluxograma ilustrado na Figura 10.

Algoritmo 1 – *Variable Neighborhood Search* - VNS

```
1 função VNS(S_0)
2  s ← S_0;
3  k ← 1;
4  r ← 3;
5  enquanto (k ≤ r) faça
6       Faça busca local em N_k(s') e obtenha s''
7        se f(s'') < f(s) então
8              s ← s''
9              k ← 1
10      senão k ← k + 1;
11      fim-se;
12 fim-enquanto;
13 retorne s;
14 fim VNS.
```

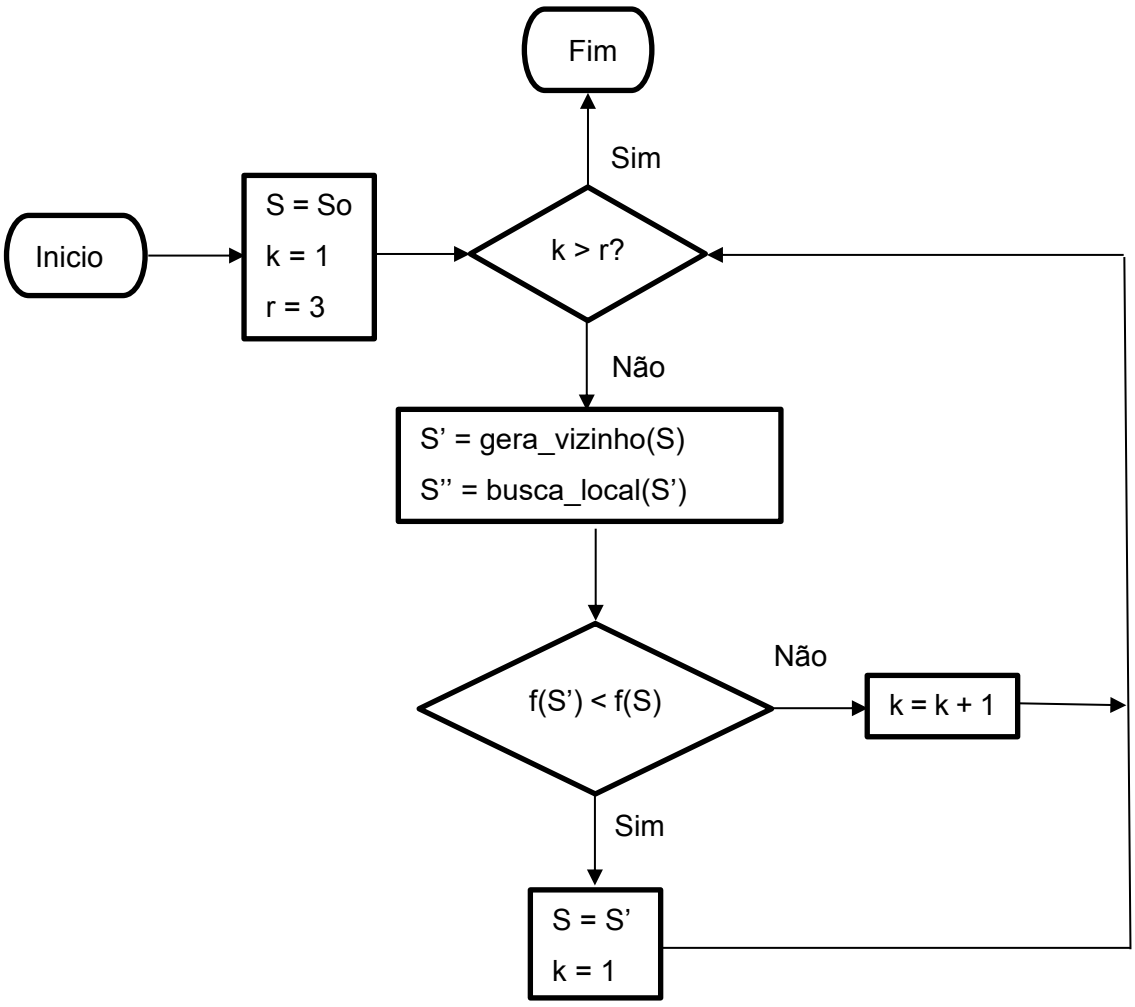
Fonte: Adaptado (Mladenovic e Hansen, 1997).

A essência do método VNS consiste em buscar a melhora da solução atual utilizando uma estrutura de vizinhança principal. O método troca a vizinhança corrente pela próxima vizinhança quando a melhora não é mais possível na vizinhança atual,

e quando uma melhor solução é encontrada, o método retoma a busca na vizinhança principal.

O fim do algoritmo é dado quando não há mais a possibilidade de melhorar a solução atual após o uso de todas as estruturas de vizinhanças consideradas.

Figura 10 - Fluxograma do *Variable Neighborhood Search* - VNS



Fonte: Autor (2022).

2.6. *VARIABLE NEIGHBORHOOD DESCENT* - VND

Quando se realiza uma busca de descida no VNS, ou seja, realiza-se um movimento para o melhor resultado na vizinhança analisada, tem-se o método *Variable Neighborhood Descent* (VND) (Algoritmo 2), conforme fluxograma ilustrado na Figura 11 (MLADENOVIC; HANSEN, 1999).

Observe que no passo 6 do Algoritmo 2, a busca se dá pela melhor solução na vizinhança atual.

O VND realiza uma busca local no seu espaço de soluções através de trocas sistemáticas de estruturas de vizinhanças, não aceitando piores soluções que a solução atual, caso não encontre, a meta-heurística realizará a busca na próxima estrutura de vizinhança, caso encontre uma solução melhor que a atual ela retorna à primeira estrutura. Se não existir uma próxima estrutura de vizinhança, a busca é interrompida e a solução atual é retornada como solução ótima local. (MLADENOVIC; HANSEN, 1997).

Algoritmo 2 - *Variable Neighborhood Descent* - VND

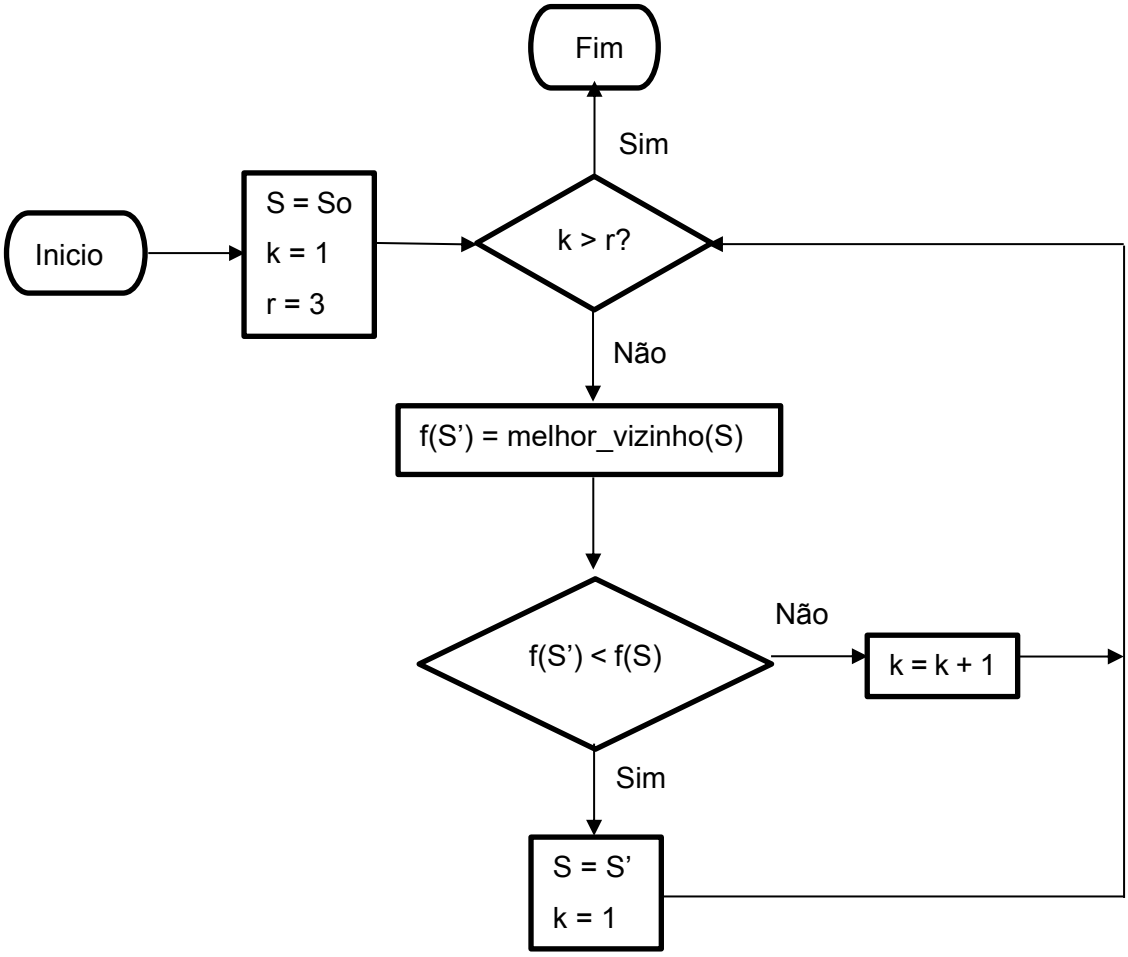
```
1 função VND(S_0)
2  s ← S_0;
3  k ← 1;
4  r ← 3;
5  enquanto (k ≤ r) faça
6       Encontre o melhor vizinho s' ∈ N_k(s);
7       se f(s') < f(s) então
8            s ← s';
9            k ← 1;
10      senão k ← k + 1;
11      fim-se;
12  fim-enquanto;
13 retorne S;
14 fim VND;
```

Fonte: Adaptado (Mladenovic e Hansen, 1997).

Pode-se observar na Figura 11, que dada uma solução inicial S_0 na função VND, atribui-se esse valor a variável da função referente a solução atual s, e os demais parâmetros k e r, para indicar o nível da vizinhança corrente ao número de vizinhanças, respectivamente. Aplica-se então um laço de repetição, que enquanto o contador das vizinhanças for menor ou igual a quantidade total de vizinhanças, realiza-se uma busca gulosa de descida para determinar a melhor solução vizinha em cada nível de vizinhança. Caso esse vizinho seja melhor que a solução atual, atribui-se essa nova solução a solução atual e volta para o nível de estrutura de vizinhança inicial.

Caso não haja melhoria na solução atual, passa-se para a próxima estrutura de vizinhança, até que a última estrutura de vizinhança seja alcançada. No final do processo o algoritmo retorna a melhor solução visitada.

Figura 11 – Fluxograma do *Variable Neighborhood Descent* - VND



Fonte: Autor (2022).

As meta-heurísticas VNS e VND têm sido bastante utilizadas em problemas de *scheduling*. Por exemplo em (TAVARES; BASTOS; REIS, 2020), os autores estudam o problema de *Flow Shop Scheduling Problem* (PFSP), no qual buscam sequenciar um conjunto de tarefas em um conjunto de, com o objetivo de minimizar o tempo total de execução do processo. Os autores propõem uma meta-heurística baseada no *Variable Neighborhood Descent* (VND) e *Iterated Local Search* (ILS) para solucionar o problema. Utilizam como busca locais o *insertion, swap* e *interchange*. Para a construção da solução inicial, utilizou-se um algoritmo aleatorizado e o NEH, proposto por (NAWAZ; ENSCORE; HAM, 1983), e as buscas locais foram incorporadas nas

meta-heurísticas para refinamento de resultados. Os métodos foram aplicados ao conjunto de instâncias proposto por (TAILLARD, 1993) e tiveram seus respectivos desempenhos comparados às melhores soluções conhecidas na literatura a fim de conferir sua eficácia.

Em Trinidad e Taglialenha (2019) estuda-se o Problema de Programação de Tripulação (PPT) de ônibus urbano, no qual consiste em determinar o número mínimo de tripulantes e especificar suas viagens de tal forma a cobrir todas as viagens da frota em operação com o menor custo possível e atendendo as restrições relacionadas à operação e às normas trabalhistas da categoria. Aplica-se uma heurística baseada em VNS para resolver o problema programação de tripulação (*Crew Scheduling Problem*) de ônibus urbanos. O método foi implementado em C++ em problemas disponíveis na literatura e os resultados obtidos foram considerados satisfatórios.

A VNS e VND também é utilizada em Problema de Roteamento de Veículos com Janela de Tempo e Múltiplos Entregadores (PRVJTME). Assunção (2020) considera um problema no qual os veículos não precisam parar necessariamente no cliente, eles param em pontos específicos próximos a eles, e a entrega é realizada a pé ou em veículo de menor porte. O autor aborda uma extensão do problema citado no qual os clientes podem tanto coletar, quanto entregar o produto, e também a frota de veículos é composta por veículos de classes diferentes. Para resolver o problema o autor elaborou um modelo matemático em Programação Inteira Mista (PLIM) e uma meta-heurística baseada em *Random Variable Neighborhood Descent* (RVND), no qual a solução inicial consiste adicionar a cada rota de maneira gulosa o cliente não-visitado cuja adição provocará menor variação na carga total, mantendo factibilidade. Em relação às vizinhanças, foram consideradas 8 vizinhanças, sendo 2 intra-rotas, no qual consiste em utilizar uma rota da solução e realizar um movimento factível transformando essa rota em outra, e 6 inter-rotas, no qual olha para duas ou mais rotas e realiza movimentos factíveis nelas para gerar novas rotas. Os resultados mostram que a heurística encontra resultados de qualidade consistente em segundos, sem sofrer maiores perdas em performance para instâncias de tamanho grande.

Melo (2022) aborda uma extensão do *School Bus Routing Problem* (SBRP), no qual inclui a alocação de estudantes aos pontos de parada selecionados, subproblemas de localização de pontos de paradas de ônibus e subproblemas de localização de pontos de paradas de ônibus e uma escola. O estudo tem como objetivo minimizar a distância total percorrida pela frota no conjunto de períodos analisados,

considerando as restrições da capacidade dos veículos e a distância máxima que os alunos podem caminhar até as paradas de ônibus. O autor propõe um modelo matemático para resolver o problema e em seguida é proposto um algoritmo baseado nas meta-heurísticas *Variable Neighborhood Descent* (VND) e *Iterated Local Search* (ILS) além de heurísticas de inserção e remoção de paradas considerando os múltiplos períodos, denominado de *Iterated Local Search with Randomized Variable Neighborhood Descent and Remove* (ILS-RVND-R). O algoritmo proposto foi executado para 95 instâncias do SBRP, propostas por Schittekat et al. (2013), e para 192 instâncias do MP-SBRP obtidas através de uma extensão das instâncias de Schittekat et al.

Mou (2022) estuda o caso de abordar o problema integrado de lotes de pedidos, sequenciamento de lotes e agendamento de separadores para minimizar o atraso total em uma loja de varejo. Para resolver o problema, o autor utiliza um modelo matemático de programação linear inteira mista e uma heurística mista combinada entre o algoritmo genético e a *Variable Neighborhood Descent*. Para a solução inicial do AG, foi introduzido um algoritmo de data de vencimento mais cedo modificado (MEDD) para gerar cromossomos de boa qualidade. Para as estruturas de vizinhança da VND, foi utilizado: trocar dois lotes de catadores diferentes, mover um pedido de cliente para um lote diferente do mesmo selecionador, mover um pedido de cliente para um selecionador diferente, trocar duas ordens do mesmo selecionador, trocar dois pedidos de catadores diferentes e trocar duas listas de seleção de diferentes selecionadores (troca de selecionador). Foram realizadas diversas extensões do estudo para validar o desempenho da abordagem proposta em relação aos algoritmos existentes na literatura recente. Por fim, o autor cita a importância da flexibilidade da força de trabalho como uma abordagem econômica para melhorar o desempenho do atendimento de pedidos na loja.

Na próxima seção será apresentada os métodos de solução propostos.

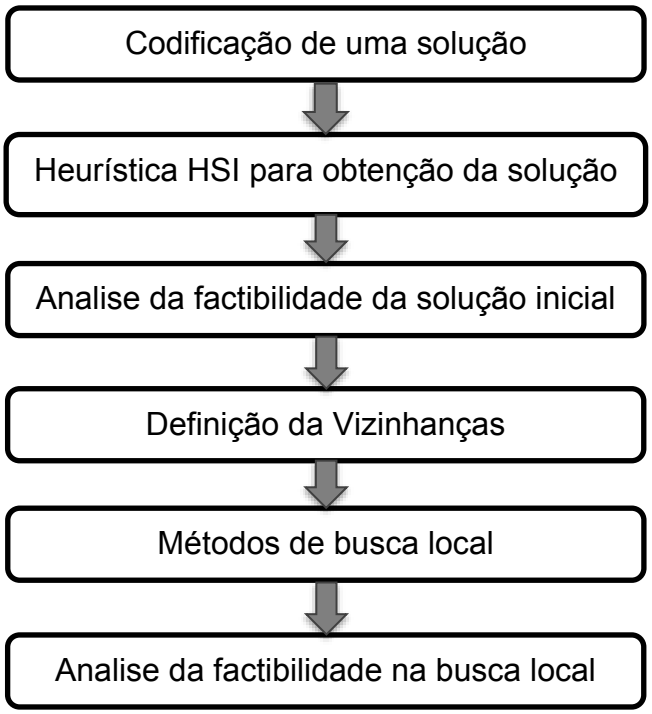
3. MÉTODOS DE SOLUÇÃO PROPOSTOS

Neste capítulo apresentam-se dois métodos de solução para o problema. O primeiro, baseado na meta-heurística VNS (MLADENOVIC; HANSEN, 1997), aqui denominado VNS-AMR. O segundo, baseado na meta-heurística VND (MLADENOVIC; HANSEN, 1997), aqui denominado VND-AMR.

3.1. DETALHAMENTO DOS ALGORITMOS PROPOSTOS

Destaca-se que durante a etapa de pesquisa bibliográfica e fundamentação teórica não foi encontrado modelo de programação matemática para determinar a do problema de *scheduling* ao se considerar dois ou mais robôs. Por isso, considerou-se o desenvolvimento de um algoritmo heurístico, cujas etapas estão listadas no fluxograma ilustrado na Figura 12, e melhor detalhadas nas seções a seguir.

Figura 12 - Etapas do método de solução



Fonte: Autor (2022).

3.1.1. Codificação da solução

Para facilitar a aplicação dos métodos heurísticos propostos utiliza-se uma codificação da solução, conforme ilustrado na Figura 13.

Considera-se uma representação da solução pela utilização de uma matriz de ordem $(4 \times n)$, na qual a primeira linha representa a sequência de máquinas que as tarefas são representadas, a segunda linha indica qual subtarefa da respectiva tarefa, a terceira linha é a rota que a ser realizada, e por fim, a última linha indica qual AMR é escolhido. Sendo n o número total de tarefas, $S(n)$ o número total de subtarefas de tarefas, r a quantidade de rotas e amr o robô escolhido.

Figura 13 - Representação da solução

Tarefa →	0	1	4	2	3	...	n
Subtarefa →	0	1	1	1	1		S(n)
Rota →	0	1	2	1	2		r
Robô →	1,2	1	2	1	2		robo

Fonte: Autor (2022).

A escolha dessa representação deu-se pela simplicidade de interpretação e pela facilidade de realizar as trocas de vizinhanças da solução durante o processo de buscas, bem como a avaliação da qualidade da solução (FO).

3.1.2. Método heurístico para a solução inicial - HSI

Utilizou-se de uma heurística construtiva gulosa controlada para determinar uma solução inicial, aqui denominada HSI. A HSI realiza os cálculos iniciais para obter as informações de horário de liberação da subtarefa k da tarefa i (e_{ik}) e do prazo limite da subtarefa k da tarefa i (d_{ik}). Com essas informações, a heurística seleciona a primeira subtarefa a ser liberada para realização. Caso tenha mais de uma subtarefa que inicie no mesmo instante, é escolhida a que tiver menor prazo de vencimento, ou seja, data de vencimento mais próximo.

Figura 14 - Exemplo de solução inicial

Tarefa →	0	1	2	3	4	1	4
Subtarefa →	0	1	1	1	1	2	2
Rota →	0	1	2	1	2	3	4
Robô →	1,2	1	2	1	2	1	2

Fonte: Autor (2022).

Na Figura 14 ilustra-se um exemplo de solução inicial, onde pode-se observar que ambos os robôs processam a tarefa 0 no armazém, a seguir o robô 1 realiza a subtarefa 1 da tarefa 1, na mesma rota o robô 1 realiza a subtarefa 1 da tarefa 3, ele volta para o armazém, reabastece, e volta a realizar as operações na rota 3, realizando a subtarefa 2 da tarefa 1. O mesmo procedimento é realizado pelo robô 2.

3.1.3. Análise de factibilidade da solução inicial

Antes de avaliar a qualidade da solução (FO da Equação (18)), é necessário verificar se a solução é factível ou não, ou seja, se atende todas as restrições do modelo dado pelas Equações (2)-(17). Caso não atenda, a solução é descartada automaticamente e uma nova tentativa de obtenção de solução é realizada.

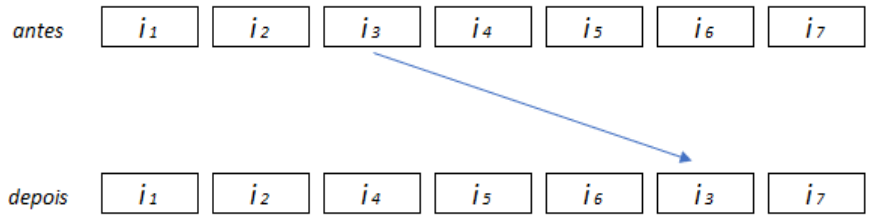
Caso a solução obtida seja factível, então a qualidade da solução é avaliada levando em conta o processamento de todas as tarefas e subtarefas da sequência de tarefas indicadas na solução. Ou seja, é levado em conta o tempo percorrido entre o armazém e os alimentadores, o tempo percorrido entre alimentadores, o tempo de processamento das tarefas, o tempo de processamento no armazém e também o tempo em que o robô fica parado esperando o horário de liberação das tarefas.

3.1.4. Definição das vizinhanças para realização das buscas locais

Para a aplicação das buscas de descida nas vizinhanças da solução de trabalho são considerados três diferentes tipos de estruturas de vizinhanças (perturbação nos parâmetros da solução), como listadas a seguir:

- *Internal Insertion*: Segundo (ZOBOLAS; TARANTILIS; IOANNOU, 2009), essa estrutura extrai uma tarefa em uma posição qualquer do vetor solução e a reinsere em uma nova posição randomicamente. (Figura 15).

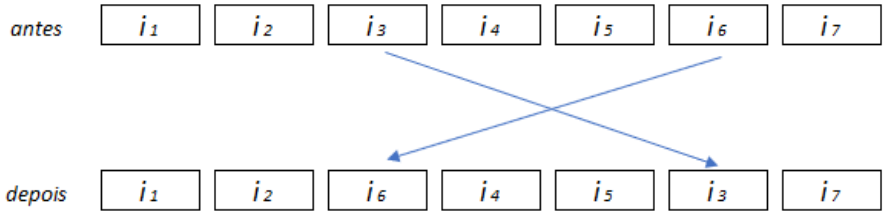
Figura 15 - *Internal Insertion*



Fonte: Autor (2022).

- *Internal Swap*: Esta estrutura troca a posição de duas tarefas diferentes aleatoriamente (ZOBOLAS et al., 2009). (Figura 16).

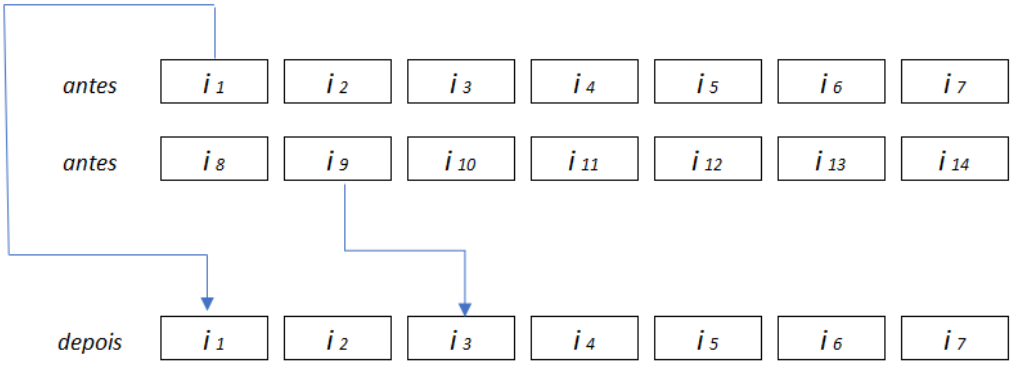
Figura 16 - *Internal Swap*



Fonte: Autor (2022).

- *External Insertion*: Com a utilização de dois vetores soluções, extrai uma posição de cada vetor solução e a insere em novo vetor, gerando uma possível nova solução. (TON, 2020) (Figura 17)

Figura 17 - *External Insertion*



Fonte: Autor (2022).

Se a tentativa de troca satisfazer as restrições impostas para o problema, a troca é efetivada, caso contrário, a troca é descartada e então escolhe-se novas posições de tarefas para troca de posição.

3.1.5. Métodos de busca local

Considerando-se a solução inicial obtida com a HSI, aplicaram-se buscas locais nas diferentes estruturas de vizinhança definidas na seção 3.1.4, conforme o algoritmo VNS e o VND.

Quando se considerou a busca segundo a meta-heurística VNS (MLADENOVIC; HANSEN, 1997), tem-se o Algoritmo VNS-AMR. E quando se considera a busca segundo a meta-heurística VND (MLADENOVIC; HANSEN, 1997), tem-se o Algoritmo VND-AMR.

3.1.5.1 Busca local no algoritmo VNS-AMR

Quando a busca local é de descida, ou seja, quando se analisam todas as soluções na vizinhança da solução atual e realiza-se um movimento aleatório para encontrar uma melhor solução, tem-se o VNS-AMR (Algoritmo 3).

Algoritmo 3 - Algoritmo VNS-AMR

1 **função** VNS-AMR(S_0)
2 $s \leftarrow S_0$;
3 $k \leftarrow 1$;
4 $r \leftarrow 3$;
5 **enquanto** $(k \leq r)$ **faça**
6 Gere uma solução aleatória s' em $N_k(s)$ e faça uma busca local em $N_k(s')$ e obtenha s'';
7 **se** $f(s'') < f(s)$ **então**
8 $s \leftarrow s''$
9 $k \leftarrow 1$
10 **senão** $k \leftarrow k + 1$;
11 **fim-se**;
12 **fim-enquanto**;
13 **retorne** s;
14 **fim** VNS-AMR.

Fonte: Autor (2022).

3.1.5.2 Busca local no algoritmo VND-AMR

Quando a busca local é de descida, ou seja, quando se analisam todas as soluções na vizinhança da solução atual e realiza-se o movimento para a melhor solução, tem-se o VND-AMR (Algoritmo 4).

Algoritmo 4 - Algoritmo VND-AMR

```
1 função VND-AMR(S_0)
2  s ← S_0;
3  k ← 1;
4  r ← 3;
5  enquanto (k ≤ r) faça
6       Encontre o melhor vizinho s' em N_k(s) ;
7       se f(s'') < f(s) então
8            s ← s''
9            k ← 1
10      senão k ← k + 1;
11      fim-se;
12 fim-enquanto;
13 retorne s;
14 fim VND-AMR.
```

Fonte: Autor (2022).

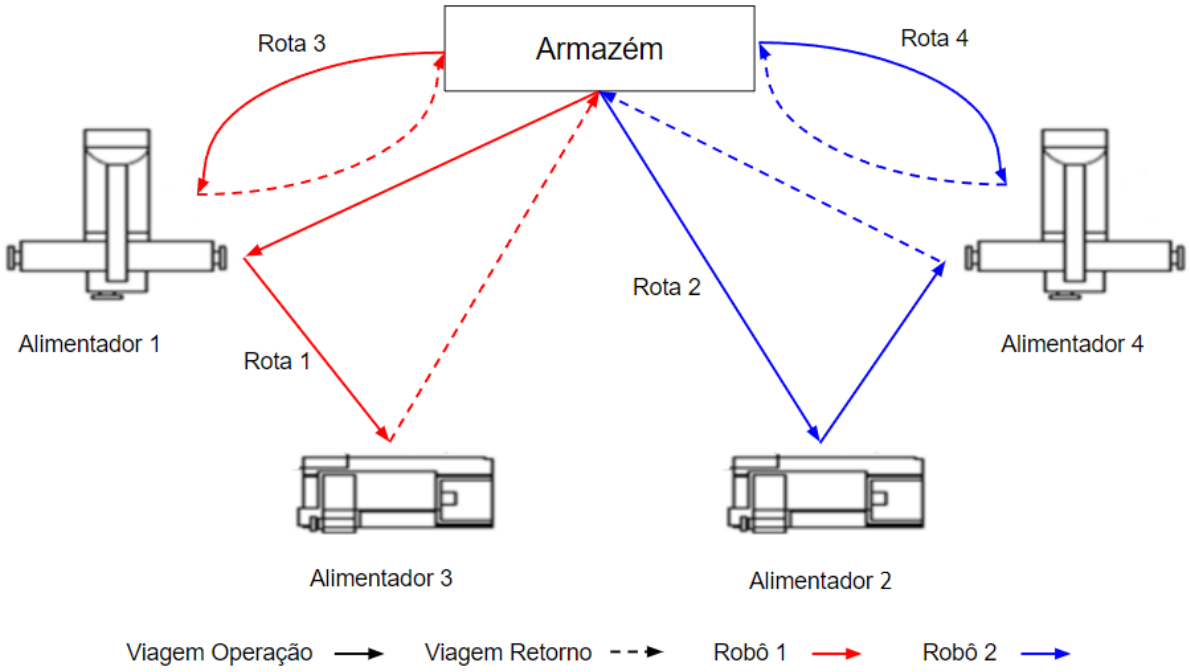
3.1.6. Análise de factibilidade da solução na busca local

Para verificar a viabilidade de uma solução durante a busca local nos algoritmos propostos VNS-AMR e VND-AMR, além de verificar o atendimento das Restrições (2)-(17), foi acrescentado uma restrição para a otimizar a FO utilizando dois robôs, no qual analisa se o tempo atual do robô somando com o tempo de ida ao armazém, abastecimento e ida ao alimentador da próxima tarefa, for menor que o tempo de liberação da próxima tarefa, o robô volta ao armazém antes de realizar a operação mesmo se ainda estiver com capacidade para processar a tarefa.

A Figura 18 ilustra o exemplo da solução da Figura 14, considerando a capacidade do robô como 3 SLC, no qual o robô 1, apresentado na figura na cor vermelha, faz a operação na subtarefa 1 da tarefa 1, então verifica se tem a capacidade suficiente para a próxima operação e se o tempo de retorno ao armazém

somando seu abastecimento mais o tempo de volta é inferior ao horário de liberação da operação, neste caso não é, portanto é realizado o processamento da operação logo em seguida, porém na próxima operação o tempo de liberação é longo, então o robô 1 volta ao armazém, realiza o abastecimento e vai no local da próxima operação para realizar o processamento na rota 3, ficando apto a realizar um conjunto de operações caso o tempo de liberação delas sejam próximas.

Figura 18 - Representação da solução



Fonte: Autor (2022).

A restrição apresentada no exemplo da Figura 18 tende a diminuir o tempo ocioso dos robôs, restringindo a área de solução, de modo que o robô fique sempre apto (abastecido) para realizar operações em subtarefas de tarefas próximas ou iguais ao seu horário de liberação.

3.2. DESTAQUE DAS PRINCIPAIS CONTRIBUIÇÕES DO MODELO PROPOSTO

Neste trabalho apresenta-se duas funções objetivos, descritas na Equação 18, para robôs homogêneos, e na Equação 19, para robôs heterogêneos, que consideram o tempo de viagem do robô entre as máquinas, o tempo de trabalho e o tempo de retorno para o armazém, diferente de Dang et al. (2013) que utiliza a

Equação 4, que considera apenas o tempo total dos deslocamentos dos robôs entre as máquinas. Outra contribuição é em relação a quantidade de robôs considerados no estudo de caso, no Dang et al. (2013) é considerado apenas um robô, enquanto que neste estudo são considerados dois robôs. No qual a definição de escolha dos robôs para processar as tarefas é feita de acordo com o horário de liberação do robô, ou seja, a primeira tarefa inicial no armazém representa o abastecimento dos robôs, para que eles fiquem aptos a realizar as operações, e é realizada por ambos os robôs no mesmo instante de tempo, a próxima tarefa é realizada pelo robô 1, e a partir daí o robô que tiver o menor tempo de operação é escolhido para realização da próxima tarefa.

No próximo capítulo apresentam-se os dados utilizados neste trabalho junto com os resultados obtidos com a aplicação dos métodos de solução aqui propostos.

4. RESULTADOS

Os dados considerados neste estudo são os dados apresentados em Dang et al. (2013). Os dados indicam os níveis máximos e mínimos de peças em cada alimentador e suas taxas de alimentação, conforme a Tabela 1.

Tabela 1 - Níveis das peças, taxa de alimentação e tempo de trabalho

		Alimentadores			
Localização	**Armazém**	**1**	**2**	**3**	**4**
Nível máximo de peças	-	250	2000	2000	250
Nível mínimo de peças	-	125	900	900	125
Taxa de alimentação (segundos/peça)	-	4,5	1,5	1,5	4,5
Tempo de trabalho do robô (segundos)	90	42	42	42	42

Fonte: Baseado em Dang et al. (2013).

Com os dados da Tabela 1 é possível calcular a periodicidade, o horário de liberação da subtarefa k da tarefa i, e o prazo limite da subtarefa k da tarefa, com as Equações (1), (2) e (3), respectivamente.

Tabela 2 - Tempo (segundos) de viagem do robô entre localidades

		Alimentadores			
Localização	**Armazém**	**1**	**2**	**3**	**4**
Armazém	0	34	37	34	40
Alimentador 1	39	0	17	34	50
Alimentador 2	35	17	0	35	49
Alimentador 3	34	33	35	0	47
Alimentador 4	36	47	48	46	0

Fonte: Baseado em Dang et al. (2013).

Com esses parâmetros, definem-se as janelas de tempo. Na última linha da Tabela 1, ilustram-se os tempos de trabalho do robô nos alimentadores e no armazém central.

Na Tabela 2 ilustram-se os tempos de locomoção entre os pares de origem e destino do robô no ambiente da fábrica.

Tabela 3 - Resultados obtidos pelo método proposto

Cenários	Quantidade de alimentadores	Cap. do robô	Total de subtarefas de tarefas	VND (s)	VNS pior (s)	VNS melhor (s)	VNS média (s)
1	2	2	4	604,50	604,50	604,50	604,50
2	2	3	4	604,50	604,50	604,50	604,50
3	3	2	6	1.167,00	1.167,00	1.167,00	1.167,00
4	3	3	6	1.167,00	1.167,00	1.167,00	1.167,00
5	4	2	8	1.167,00	1.167,00	1.167,00	1.167,00
6	4	3	8	1.167,00	1.167,00	1.167,00	1.167,00
7	4	2	10	1.729,50	1.729,50	1.729,50	1.729,50
8	4	3	10	1.729,50	1.729,50	1.729,50	1.729,50

Fonte: Autor (2022).

É possível observar na Tabela 3 os diferentes cenários estudados, e os resultados encontrados em segundos, de cada cenário, para os dois métodos. Para ser considerada uma solução viável deve-se abastecer o alimentador completamente, caso não seja possível, a solução é descartada.

Pode haver diversos conjuntos de operações para o mesmo número do total de subtarefas de tarefas, como por exemplo, o cenário 8 pode ser representado por 3 subtarefas para se completar a tarefa 1, 2 subtarefas para se completar a tarefa 2, 2 subtarefas para se completar a tarefa 3 e por 3 subtarefas para se completar a tarefa 4, resultando em 10 subtarefas, e também pode ser representado por, 4 subtarefas para se completar a tarefa 1, 1 subtarefa para se completar a tarefa 2, 1 subtarefa para se completar a tarefa 3 e por 4 subtarefas para se completar a tarefa 4, resultando em 10 subtarefas. Esse último exemplo foi utilizado no trabalho.

Destaca-se que o modelo proposto neste estudo considera a utilização de dois robôs e duas meta-heurísticas de melhoria. Nota-se que os valores das meta-heurísticas ficaram idênticas, isso acontece devido qualidade da solução inicial, que escolhe a subtarefa com o menor horário de liberação.

Para verificar a qualidade das meta-heurísticas desenvolveu-se uma heurística aleatória controlada para construção de uma solução inicial viável. Na Tabela 4 encontram-se os resultados dos valores da função objetivo das meta-heurísticas.

Para a construção da Tabela 4 foram consideradas 10 iterações, resultando em uma diferença média de 1,25% do tempo de conclusão de todas as tarefas em relação aos cenários estudados.

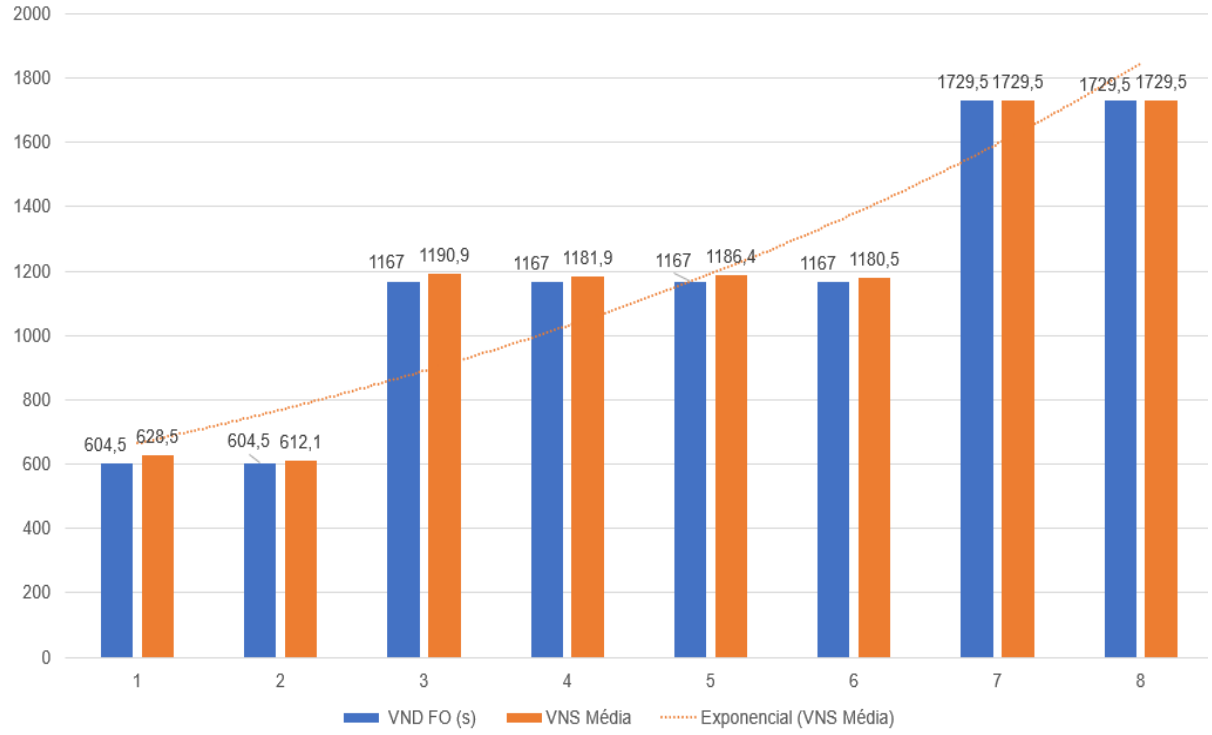
Tabela 4 - Resultados obtidos pelo método proposto com heurística aleatória

Cenários	Quantidade de alimentadores	Cap. do robô	Total de subtarefas de tarefas	VND (s)	VNS pior (s)	VNS melhor (s)	VNS média (s)
1	2	2	4	604,50	692,50	604,50	628,50
2	2	3	4	604,50	680,50	604,50	612,10
3	3	2	6	1.167,00	1.257,00	1.167,00	1.190,90
4	3	3	6	1.167,00	1.257,00	1.167,00	1.181,90
5	4	2	8	1.167,00	1.243,00	1.167,00	1.186,40
6	4	3	8	1.167,00	1.243,00	1.167,00	1.180,50
7	4	2	10	1.729,50	1.729,50	1.729,50	1.729,50
8	4	3	10	1.729,50	1.729,50	1.729,50	1.729,50

Fonte: Autor (2022).

Os cenários compreendem a variação da quantidade de alimentadores, da capacidade do(s) robô(s) e do total de subtarefa de tarefas. Conforme pode ser visto na Figura 19 em que a curva em laranja representa a VND-AMR e a curva em azul a média pela VNS-AMR.

Figura 19 - Resultados obtidos pelo método proposto



Fonte: Autor (2022).

A curva em exponencial apresentada na Figura 19 mostra que conforme aumenta a quantidade de subtarefas aumenta-se também o tempo necessário para o processamento das mesmas.

Foram divididos em dois grupos conforme a capacidade máxima utilizada, o primeiro grupo considerou a capacidade máxima utilizando dois SLC, enquanto que no segundo grupo foi considerada a capacidade máxima utilizando três SLC, no qual está detalhado na Tabela 3 e Tabela 4. Ao final, cada cenário foi dividido em casos com diferentes números de tarefas e subtarefas de tarefas. O horizonte de planejamento do estudo T foi de aproximadamente 2.700 segundos (45 minutos), devido a capacidade da bateria do robô.

Ao se considerar um cenário em que se realiza uma extensão do estudo de caso, mantendo a quantidade de alimentadores igual a 4, obtém-se os resultados das soluções viáveis apresentadas na Tabela 5.

Tabela 5 - Extensão do estudo de caso

Quantidade de alimentadores	Capacidade do robô	Total de subtarefas de tarefas	VND (s)	VNS pior (s)	VNS melhor (s)	VNS média (s)
4	2	16	2.854,50	2.944,50	2.854,50	2.897,65
4	3	16	2.854,50	2.942,50	2.854,50	2.884,40
4	2	32	6.229,50	6.319,50	6.229,50	6.278,60
4	3	32	6.229,50	6.319,50	6.229,50	6.260,80

Fonte: Autor (2022).

4.1. ANÁLISE DOS RESULTADOS

É possível observar pelos resultados da Tabela 5, que a meta-heurística baseada em VND e VNS propostas, encontram soluções viáveis para problemas de grande porte, ao passo que ao se utilizar o PLIM, não é possível encontrar uma solução para o problema devido a sua característica *NP-Hard*. E à medida que o tamanho do problema aumenta, especialmente o número de subtarefas de tarefas, o tempo de solução da meta-heurística proposta se torna maior.

No próximo capítulo são apresentadas as conclusões sobre o estudo, modelos escolhidos, dados utilizados, e sugestões para trabalhos futuros sobre o tema apresentado

5. CONCLUSÕES

Este trabalho apresenta o resultado do estudo do problema de *scheduling* de dois robôs móveis autônomos, responsáveis por processar tarefas de alimentação de peças em máquinas em um ambiente de manufatura flexível, no qual se definem restrições de janela de tempo para a realização das tarefas de alimentação baseado no sistema de inventário de estoque (s, Q), e leva em consideração os atributos das máquinas, como a taxa de alimentação (peça/segundos), de modo a minimizar o tempo total para processar todas as tarefas utilizando os robôs móveis autônomos.

Vale destacar que o problema de *scheduling* é um problema *NP-Hard*, portanto, só é possível utilizar solução via PLIM para casos com poucas tarefas e poucos alimentadores. Na revisão realizada para este estudo, observou-se que existe apenas modelos que considera um robô. Este trabalho propõe duas novas FOs que considera dois ou mais robôs, para o problema de AMR, uma que considera robôs iguais e outra com robôs com características diferentes. É proposto dois métodos meta-heurísticos baseados em trocas sistemáticas de estruturas de vizinhança, um baseado no VND e outro no VNS.

Para casos maiores só é possível obter soluções com a utilização de métodos aproximativos, como os métodos meta-heurístico baseados em trocas sistemáticas de estruturas de vizinhança considerados nesse estudo, que possuem a habilidade de escapar dos ótimos locais, podendo encontrar soluções viáveis de boa qualidade para problemas de médio e grande porte considerando dois robôs. Além do método ser facilmente adaptado para tratar casos reais, sendo necessário apenas alterar os parâmetros de capacidade e tempo de percurso dos robôs aos alimentados e ao deposito, e modificar o tempo de processamento dos alimentadores e do deposito, além de alterar a quantidade de alimentadores existentes na área de manufatura estudada.

O método foi implementado em linguagem Python, onde se pode concluir que construir uma solução inicial viável buscando o menor horário de liberação da subtarefa tende a encontrar uma solução de boa qualidade para o problema de *scheduling* junto a meta-heurística VND, porém a VNS, por ser estocástica, ou seja,

busca-se uma solução oriunda de um vizinho aleatório, ela tende a ter maior dificuldade de encontrar uma solução de boa qualidade, como a solução encontrada pela VND, resultando em uma diferença média de 1,25% de tempo de conclusão de todas as tarefas em relação ao cenário base estudado ilustrado na Tabela 4.

O estudo desenvolvido neste Trabalho de Conclusão de Curso proporcionou um maior conhecimento acadêmico e profissional, permitindo aplicar na prática os conhecimentos adquiridos na academia em uma área de grande relevância acadêmica.

5.1. TRABALHOS FUTUROS

Para estudos futuros recomenda-se:

- Propor um modelo PLIM para considerar dois ou mais robôs, para verificação dos resultados;
- Definir e implementar novas estruturas de vizinhança;
- Considerar uma FO com robôs com tempos de processamentos diferentes;
- Implementar os modelos proposto em casos reais.

REFERÊNCIAS

ABB lista as principais tendências que mudarão a automação robótica. **InforChannel**, 2022. Disponível em: < https://inforchannel.com.br/2022/03/21/abb-lista-as-principais-tendencias-que-mudarao-a-automacao-robotica/>. Acesso em: 17 dez. 2022.

ANAND, R.; AGGARWAL, D.; KUMAR, V. A comparative analysis of optimization solvers. **Journal of Statistics and Management Systems**, Taylor Francis, v. 20, n. 4, p. 623–635, nov. 2017. Disponível em: https://doi.org/10.1080/09720510.2017.1395182.

ANDRADE, F. V. **Estudo comparativo aplicado ao planejamento e controle de trajetórias de robôs móveis**. 2011. Dissertação (Mestrado em Engenharia Elétrica) – Centro de Tecnologia, Universidade Federal do Ceará, Fortaleza, 2011.

ARENALES, M., ARMENTANO, V., MORABITO, R., YANASSE, H. **Pesquisa Operacional**. Rio de Janeiro: Elsevier: ABEPRO, 2011.

ASSUNÇÃO, M. A. F. **Problema de roteamento de veículos heterogêneos com múltiplos entregadores e coleta e entrega simultâneas**. 2020. Dissertação (Mestrado em Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, São Paulo, Universidade de São Paulo, 2020.

BAKER, K. R. **Introduction to sequencing and scheduling**. New York: Wiley, 1974.

BARBOSA, L. B. **Otimização do sequenciamento de tarefas em máquinas paralelas com tempos de processamento diferente**. 2017. Trabalho de Conclusão de Curso (Graduação em Engenharia de Transportes e Logística) – Centro Tecnológico de Joinville, Universidade Federal de Santa Catarina, Joinville, 2017.

BOWERSOX, D. J.; CLOSS, D. J. **Logística empresarial:** o processo de integração da cadeia de suprimentos. São Paulo: Atlas, 2011.

BRANQUINHO, V. M. F. **Escalonamento de produção com tempos de setup dependentes** – Aplicação na SAPEC Agro. 2013. Dissertação (Mestrado em Decisão Económica e Empresarial) - Lisbon School of Economics & Management, Lisboa, 2013.

COLIN, E. C. **Pesquisa Operacional:** 170 aplicações em estratégia, finanças, logística, produção, marketing e vendas. Rio de Janeiro: LTC, 2007.

CUI, M.; BAI, R.; LU, Z., LI, X.; AICKELIN, U.; GE, P. Regular expression based medical text classification using constructive heuristic approach. **IEEE Access,** v. 7, p. 147892-147904, out. 2019.

DANG, Q. V.; STEGER-JENSEN, I. N. K.; MADSEN, O. Scheduling a single mobile robot for part-feeding tasks of production lines. **Journal Of Intelligent Manufacturing**, v. 25, p. 1271–1287, 2013.

DANG, Q. V.; NGUYEN, C. T.; RUDOVÁ, H. Scheduling of mobile robots for transportation and manufacturing tasks. **Journal of Heuristics**, v. 25, p. 175–213, 2019.

DIAS, S. V. E. **Otimização da programação de produção em ambiente job shop através da minimização do makespan:** um estudo de caso em uma indústria de cosméticos. 2015. Monografia (Especialização em Engenharia de Produção) – Centro Politécnico, Universidade Federal do Paraná, Curitiba, 2015.

FAZLOLLAHTABAR, H.; SAIDI-MEHRABAD, M. Method-ologies to optimize automated guided vehicle scheduling and routing problems: a review study. **Journal of Intelligent & Robotic Systems**, v. 77, p. 525–545. 2015

FERNANDES, H. A. **Simulated annealing para programação de robôs móveis autônomos em ambiente de manufatura flexível**. 2021. Trabalho de Conclusão de Curso (Graduação em Engenharia de Transportes e Logística) – Centro Tecnológico de Joinville, Universidade Federal de Santa Catarina, Joinville, 2021.

FERNANDES, H. A.; TAGLIALENHA, S.L.S. Sequenciamento de tarefas e agendamento de robôs móveis autônomos. In: XLI ENCONTRO NACIONAL DE ENGENHARIA DE PRODUÇÃO, 2021, Foz do Iguaçu. **Anais do Encontro Nacional de Engenharia de Produção - Enegep**, 2021. v. 1.

FILHO, W.M. **Desenvolvimento e aplicação de algoritmos heurísticos ao problema de alocação de espaço físico**. 2008. Dissertação (Mestrado em Ciência da Computação) - Universidade Estadual de Maringá, Maringá, 2008.

FISCHETTI, M.; MARTELLO, S.; TOTH, P. The fixed job schedule problem with spread-time constraints. **Operations Research**, v. 35, n. 6, p. 849–858, 1987.

FLIZICOSKI, A. L. V. **Aplicação da minimização do atraso total em ambiente de máquina única com tempos de setup dependentes da sequência.** 2017. Trabalho de Conclusão de Curso (Graduação em Engenharia de Produção) – Departamento de Engenharia de Produção, Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2017.

GIL, A. C. **Como elaborar projetos de pesquisa**. 6. Ed. São Paulo: Atlas, 2019.

GOLDBERG, D. E. Genetic algorithms in search**. Optimization, and Machine Learning**, 1989.

HILLIER, F. S.; LIEBERMAN, G. J. **Introdução à pesquisa operacional**. 9. ed. Porto Alegre: AMGH, 2013.

HOCHBAUM, D. S. The Scheduling Problem. **RIOT: Remote Interactive Optimization Testbed**, dez. 1999. Disponível em: https://riot.ieor.berkeley.edu/Applications/Scheduling/l. Acesso em novembro de 2022.

KIM, J.Y.; CHO, J.W.; KWON, J.H. Multi-Robot Traffic Management using MIP Path Negotiation Scheduler. **21st International Conference on Control, Automation and SYSTEMS (ICCAS)**, p. 2196-2199, dez. 2021. Disponível em: https://ieeexplore.ieee.org/document/9649860. Acesso em novembro de 2022.

JOO, C. M.; KIM, B. S. Machine scheduling of time-dependent deteriorating jobs with determining the optimal number of rate modifying activities and the position of the activities. **Journal of Advanced Mechanical Design, Systems, and Manufacturing**, v. 9, n. 1, mar. 2015.

JUN, S.; LEE, S.; YIH, Y. Pickup and delivery problem with recharging for material handling systems utilising autonomous mobile robots. **European Journal of Operational Research**, v. 289, Issue 3, p. 1153-1168, mar. 2021. https://doi.org/10.1016/j.ejor.2020.07.049. Acesso em novembro de 2022.

LAGER, A.; SPAMPINATO, G.; PAPADOPOULOS, A. V.; NOLTE, T. Task roadmaps: speeding up task replanning. Frontiers in Robotics and AI. Jun. 2022. https://doi.org/10.3389/frobt.2022.816355. Acesso em novembro de 2022.

LOUREIRO, N. F. P. **Utilização do simulated annealing na resolução de problemas no planejamento de produção**. 2014. Dissertação (Mestrado em Engenharia de Gestão Industrial) – Departamento de Engenharia Mecânica, Faculdade de Ciências e Tecnologia da Universidade de Coimbra, Coimbra, 2014.

LU, Y. Industry 4.0: a survey on technologies, applications and open research issues. **Journal of Industrial Information Integration**, v. 6, p. 1-10, jun. 2017.

MACDOUGALL, W. **Industrie 4.0: Smart Manufacturing for the Future**. Berlin, 2014.

MAIA, P. M.; BESSANI, M. Análise do efeito de incertezas de tempo aplicadas ao problema de roteamento de veículos com janelas de atendimento. **LII Simpósio Brasileiro de Pesquisa Operacional**, v. 52, nov. 2020.

MANAFI, E.; TAVAKKOLI-MOGHADDAM, R.; MAHMOODJANLOO, M. A centroid opposition-based coral reefs algorithm for solving an automated guided vehicle routing problem with a recharging constraint. **Applied Soft Computing,** v.128, ago. 2022. Disponível em: https://doi.org/10.1016/j.asoc.2022.109504. Acesso em novembro de 2022.

MELO, I. E. S. **Modelagem matemática e algoritmos heurísticos para o problema de roteamento de ônibus escolares envolvendo múltiplos períodos**. 2022. Dissertação (Mestrado em Engenharia de Produção) – Universidade Federal de Pernambuco, Recife. 2022.

Mercado de robôs móveis autônomos - crescimento, tendências, impacto do covid-19 e previsões (2022 - 2027). **Mordor Intelligence**. 2022. Disponível em: < https://www.mordorintelligence.com/pt/industry-reports/autonomous-mobile-robot-market>. Acesso em: 17 dez. 2022.

MLADENOVIC, N.; HANSEN, P. Variable neighborhood search. **Computers & operations research,** Oxford, v. 24, n. 11, p. 1097–1100, 1997.

MLADENOVIC, N.; HANSEN, P. An introduction to variable neighborhood search. **Meta-heuristics: Advances and Trends in Local Search Paradigms for Optimization,** Springer, Boston, pp. 433–458, 1999.

MOU, S. Integrated order picking and multi-skilled picker scheduling in omni-channel retail stores. **Mathematics**. 2022; 10(9):1484. Disponível em: https://doi.org/10.3390/math10091484. Acesso em novembro de 2022.

NAWAZ, M.; ENSCORE, J. R. E.; HAMI. A heuristic algorithm for the m-machine n-job flow-shop sequencing problem. Omega. **The International Journal of Management Science**, 11:91–95,1983.

NIELSEN, I. et al. Scheduling of mobile robots with preemptive tasks. **Distributed Computing and Artificial Intelligence**, Springer, Switzerland, p. 19-27, 2014.

Pacheco, R. N.; Costa, A. H. R. Navegação de robôs móveis utilizando o método de campos potenciais, in M. T. S. Sakude and C. de A. Castro Cesar (eds), **Workshop de Computação – WORKCOMP'2002**, Instituto Tecnológico de Aeronáutica, São José dos Campos, SP, pp. 125–130, 2002.

(PDF) Aprendizado da coordenação de comportamentos primitivos para robôs móveis. Available from: https://www.researchgate.net/publication/250987597_Aprendizado_da_coordenacao_de_comportamentos_primitivos_para_robos_moveis [accessed Dec 17 2022].

PAPE, C. L. **Constraint-based scheduling:** A tutorial. [S.l.], 2015.

POUDEL, L.; ZHOU, W.; SHA, Z. Resouse-Constrained Scheduling for Multi-Robot Cooperative Three-Dimensional Printing. Journal of Mechanical Desing, v. 143, n 7, jul., 2021. Disponível em: https://doi.org/10.1115/1.4050380. Acesso em novembro de 2022.

PINEDO, M. L. **Scheduling** - theory, algorithms, and systems. 3. ed. Nova Iorque: Springer, 2008.

REIS, J. Uma introdução ao scheduling. **Instituto Superior de Ciências do Trabalho e da Empresa**, Lisboa, out. 2006. Disponível em: https://repositorio.iscte-iul.pt/handle/10071/169. Acesso em novembro de 2022.

REIS, P. C. S. O. **Ferramenta de apoio ao escalonamento da produção**. 2020. Dissertação (Mestrado em Engenharia e Gestão Industrial) – Departamento de Engenharia Mecânica, Instituto Superior de Engenharia do Porto, Porto, 2020.

REKLAITIS, G. V. Review of scheduling of process operations. **Alche Symposium Series**, v. 78, n. 214, p. 119–133, 1982.

RIBEIRO, C. H. C.; COSTA, A. H. R.; ROMERO, R. A. F. Robôs Móveis Inteligentes: Princípios e Técnicas. Em A.T. Martins and D.L. Borges (editors), **Anais do XXI Congresso da Sociedade Brasileira de Computação**, SBC, Fortaleza, 2001, Vol. 3, pp.257-306.

RIBEIRO, R. Mercado global de robôs móveis autônomos deve atingir US$ 12,4 bilhões em 2030. MundoGEO, 2022. Disponível em: < https://mundogeo.com/2022/10/13/mercado-global-de-robos-moveis-autonomos-deve-atingir-us-124-bilhoes-em-2030/>. Acesso em: 17, dez. 2022.

SANTO, L. C. E. **Sequenciamento de máquinas através de Algoritmos Genéticos**. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Departamento de Computação, Universidade Estadual de Londrina, Londrina, 2014.

SANTOS, V. L. A. **Sequenciamento de tarefas em máquinas paralelas com desgastes dependentes da sequência:** resolução heurística. Dissertação (Mestrado em Ciências da Computação) - Departamento de Informática, Universidade Federal de Viçosa, Viçosa, 2016.

SHEN, W.; WANG, L. & HAO, Q. Agent-based distributed manufacturing process planning and scheduling: a state-of-the-art survey**. IEEE Transactions on Systems, Man and Cybernetics**, vol. 36, no. 4, p. 563-577, jul. 2006.

S. RIAZI, S.; LENNARTSON, B. Using CP/SMT Solvers for Scheduling and Routing of AGVs. **IEEE Transactions on Automation Science and Engineering**, vol. 18, no. 1, pp. 218-229, jan. 2021, Disponível em: https://ieeexplore.ieee.org/document/9167391. Acesso em novembro de 2022.

SCHITTEKAT, P.; KINABLE, J.; SÖRENSEN, K.; SEVAUX, M.; SPIEKSMA, F.; SPRINGAEL, J. A metaheuristic for the school bus routing problem with bus stop selection. **European Journal of Operational Research**, Elsevier, v. 229, n. 2, p. 518–528, 2013.

TAILLARD E. Benchmark for basic scheduling problems. **European Journal of Operational Research**, 64(2):278-285, 1993.

TAVARES, D. M. L.; BASTOS, L. DOS S. L.; REIS, K. A. Uma proposta de heurísticas e meta-heurísticas aplicadas ao problema de flow shop scheduling / A proposal of heuristics and metaheuristics for solving the flow shop scheduling problem. **Brazilian Journal of Development**, 6(6), 38266–38282, jun. 2020. Disponível em: https://doi.org/10.34117/bjdv6n6-390. Acesso em novembro de 2022.

THOBEN, K. D.; WIESNER, S.; WUEST, T. "Industrie 4.0" and smart manufacturing - a review of research issues and application examples. **International Journal of Automotive Technology**, v. 11, n. 1, p. 4–16, jan. 2017.

TRINIDAD, J. C. T; TAGLIALENHA, S. L. S. Método heurístico para a otimização da programação de tripulações no transporte público urbano. **Anais do IX Congresso brasileiro de Engenharia de Produção**, Ponta Grossa, 2019.

ULLRICH, G. **Automated guided vehicle systems** - a primal with practical applications. Voerde, germany: Springer, 2014, ch. 1, pp. 4–14.

VARELA, M. L. R. **Uma contribuição para o escalonamento da produção baseado em métodos globalmente distribuídos**. 2007. Tese (Doutorado em Engenharia de Produção e Sistema) – Departamento de produção e sistemas, Escola de Engenharia da Universidade do Minho, Braga, nov. 2007.

VASQUEZ, J. C. D. **Programação de tarefas em um ambiente *flow shop* com *m* máquinas para a minimização do desvio absoluto total de uma data de entrega**

comum. 2017. Dissertação (Mestrado em Ciências) – Instituto de matemática e estatística, Universidade de São Paulo, São Paulo, 2017.

VERÍSSIMO, J.M.F. **Lot Sizing and Scheduling** - A case study in the plastic Enjection Industry. Dissertação (Mestrado em Engenharia e Gestão Industrial) – Departamento de Engenharia mecânica, Faculdade de Ciências e Tecnologia da Universidade de Coimbra, Coimbra, 2016.

VIEIRA, R. C.; TAGLIALENHA, S. L. S. Heurística de vizinhança variável para programação de robôs móveis autônomos em ambiente de manufatura flexível. Anais do **XLII Encontro Nacional de Engenharia de Produção**, Foz do Iguaçu, 2022. Disponível em https://www.abepro.org.br/biblioteca/TN_ST_384_1901_43413.pdf. Acesso em outubro de 2022.

WIGHT, O. W. **Production and inventory management in the computer age**. Nova Jersey: John Wiley & Sons, Inc., 1984.

WOLF, D. F.; OSÓRIO, F. S.; Simões, E.; Trindade Jr., Onofre. 2009. **Intelligent Robotics: From Simulation to Real World Applications**. Tutorial, ICMC, LRM, USP, São Carlos, SP. Disponível em: <http://inct-sec.icmc.usp.br/actrep/sites/default/files/highlights/Tutorial-JAI.pdf>. Acesso em: 17 dez. 2022.

ZOBOLAS, G. I.; TARANTILIS, C. D.; IOANNOU, G. Minimizing makespan in permutation flow shop scheduling problems using a hybrid metaheuristic algorithm. **Computers & Operations Research**, v. 36, p. 1249-1267, 2009.