UNIVERSIDADE FEDERAL DE OURO PRETO

INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS

DEPARTAMENTO DE COMPUTAÇÃO

MATEUS FILIPE MOREIRA SILVA

Orientador: Prof. Dr. Marco Antonio Moreira de Carvalho

METAHEURÍSTICA APLICADA AO PLANEJAMENTO DA PRODUÇÃO EM SISTEMAS DE MANUFATURA FLEXÍVEL: PERÍODOS DE PRODUÇÃO NÃO SUPERVISIONADA, PROCESSAMENTO PRIORITÁRIO DE TAREFAS E CUSTOS RELACIONADOS A TROCAS DE FERRAMENTAS

Ouro Preto, MG

2025

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

MATEUS FILIPE MOREIRA SILVA

METAHEURÍSTICA APLICADA AO PLANEJAMENTO DA PRODUÇÃO EM SISTEMAS DE MANUFATURA FLEXÍVEL: PERÍODOS DE PRODUÇÃO NÃO SUPERVISIONADA, PROCESSAMENTO PRIORITÁRIO DE TAREFAS E CUSTOS RELACIONADOS A TROCAS DE FERRAMENTAS

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Marco Antonio Moreira de Carvalho

Ouro Preto, MG
2025

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E BIOLOGICAS
DEPARTAMENTO DE COMPUTACAO

FOLHA DE APROVAÇÃO

Mateus Filipe Moreira Silva

Metaheurística aplicada ao planejamento da produção em sistemas de manufatura flexível: períodos de produção não supervisionada, processamento prioritário de tarefas e custos relacionados a trocas de ferramentas

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Ciência da Computação

Aprovada em 31 de Março de 2025.

Membros da banca

Marco Antonio Moreira de Carvalho (Orientador) - Doutor - Universidade Federal de Ouro Preto
Joubert de Castro Lima (Examinador) - Doutor - Universidade Federal de Ouro Preto
André Luís Barroso Almeida (Examinador) - Doutor - Instituto Federal de Minas Gerais - Campus Ouro Preto

Marco Antonio Moreira de Carvalho, Orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 31/03/2025.

Documento assinado eletronicamente por **Marco Antonio Moreira de Carvalho**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 31/03/2025, às 16:49, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0885588** e o código CRC **F8DC136B**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: 3135591692 - www.ufop.br

Dedico este trabalho aos meus pais, Nereu e Marilene, que me deram ensinamentos valiosos e proporcionaram oportunidades que nem mesmo eles tiveram. Também dedico aos meus falecidos avós, Nereu e Carmem.

Agradecimentos

Primeiramente, agradeço ao professor Marco Antonio por toda a sua orientação e pelos ensinamentos ao longo da graduação. Ele é uma pessoa incrível, com um enorme compromisso com seus alunos, sempre disposto a ajudar em qualquer situação, sem medir esforços.

Também agradeço ao meu irmão, Lucas, que trilhou o mesmo caminho que estou seguindo, compartilhando comigo ensinamentos e conselhos valiosos. O mesmo digo aos meus grandes amigos, Rômulo, Ederson e Guilherme. Agradeço a todos pelos bons momentos que tivemos durante nossa caminhada.

Resumo

Com o aumento da demanda dos mercados consumidores por uma maior variedade de produtos em quantidades menores, as empresas têm adotado modelos de manufatura flexível, como o sistema *high-mix*, *low-volume*, para se adaptar às exigências do mercado. Esse cenário exige soluções inovadoras para otimizar o planejamento da produção e reduzir custos. Na presente monografia, propõe-se a aplicação de uma meta-heurística, especificamente o revenimento paralelo (*parallel tempering*, PT), para o planejamento da produção em sistemas de manufatura flexível. O problema abordado, conhecido como *job sequencing and tool switching problem* (SSP), envolve a otimização da sequência de tarefas, com o objetivo de minimizar o número de trocas de ferramentas, considerando também outros custos associados e períodos de produção não supervisionada. As características da versão do SSP abordada neste estudo incluem múltiplas máquinas, tarefas com diferentes prioridades e reentrância de operações, o que aumenta a complexidade do problema. A abordagem proposta foi testada em novas instâncias baseadas em dados reais de uma indústria de manufatura, com adaptações necessárias para lidar com inconsistências nos dados. Os resultados indicam que o método PT é eficaz para produzir soluções viáveis de boa qualidade para estas instâncias. Conclui-se que a aplicação do PT é uma estratégia viável para otimização em sistemas de manufatura flexível. A abordagem proposta demonstrou estabilidade e robustez, embora haja desafios a serem superados, especialmente no tempo de convergência e na eficiência do cálculo da função de avaliação. As perspectivas futuras incluem otimizações computacionais, a introdução de novos conjuntos de instâncias e a implementação de um modelo matemático comparativo, consolidando as bases para avanços na área.

Palavras-chave: Manufatura Flexível. Job Sequencing and Tool Switching Problem. Revenimento Paralelo.

Abstract

With the increasing demand from consumer markets for a greater variety of products in smaller quantities, companies have adopted flexible production models, such as the high-mix, low-volume system, to adapt to the criteria of the market. This scenario requires innovative solutions to optimize production planning and reduce costs. In this monograph, we propose the application of a metaheuristic, precisely parallel tempering (PT), for production planning in flexible production systems. The problem addressed, the job sequencing and tool switching problem (SSP), involves optimizing the task sequence to minimize the number of tool changes and consider other associated costs and unsupervised production periods. The characteristics of the SSP version addressed in this study include multiple machines, tasks with different priorities, and reentrant operations, increasing the problem's complexity. The proposed approach was tested on new instances based on real data from a production industry, with adaptations allowed to deal with inconsistencies in the data. The results indicate that the PT method effectively produces good quality feasible solutions for these instances. It is concluded that applying PT is a viable strategy for optimization in flexible production systems. The proposed approach demonstrated stability and robustness, although there are challenges to be overcome, especially regarding convergence time and the efficiency of the evaluation function calculation. Future perspectives include computational optimizations, the introduction of new instance sets, and the implementation of a comparative mathematical model, laying the foundation for advancements in the field.

Keywords: Flexible Manufacturing. Job Sequencing and Tool Switching Problem. Parallel Tempering.

Lista de Ilustrações

Lista de Tabelas

Lista de Algoritmos

Lista de Abreviaturas e Siglas

FMS	*flexible manufacturing system*
GPCA	*greedy pipe construction algorithm*
HMLV	*high-mix, low-volume*
KTNS	*keep tool needed soonest*
MCMC	Monte Carlo Markov Chain
PT	*parallel tempering*
SSP	*job sequencing and tool switching problem*

Sumário

1 Introdução

Com o advento da indústria 4.0, também referida como quarta Revolução Industrial, emergem novos paradigmas para os processos produtivos, desencadeando uma série de desafios e oportunidades (Morgan et al., 2021). Adicionalmente, as crescentes demandas do mercado, tais como a necessidade de diversificação de produtos e a pressão por redução de custos, impulsionam a busca incessante por soluções inovadoras e disruptivas (Fogliatto; da Silveira; Borenstein, 2012). Como consequência, a interdependência entre processos digitais e físicos assume um papel central, procurando otimizar a interação entre sistemas computacionais e componentes de controladores físicos da melhor forma possível (Splunk, 2024). Nesse contexto, as empresas enfrentam o desafio de se adaptar rapidamente às mudanças tecnológicas e às demandas do mercado, a fim de se manterem competitivas e sustentáveis a longo prazo.

A emergência da demanda do mercado consumidor por uma ampla gama de produtos tem impulsionado a adoção de um modelo de manufatura conhecido como *high-mix*, *low-volume* (HMLV) (Gan; Musa; Yap, 2023). Esse paradigma responde à necessidade de produzir uma variedade cada vez maior de produtos em quantidades menores. No HMLV, a fabricação ocorre em lotes pequenos, permitindo a flexibilidade necessária para atender às demandas do mercado em constante mudança. Cada produto único implica em tempos de produção distintos e ajustes específicos de maquinário, tornando a programação da produção um desafio complexo. Gerenciar a alocação de recursos e determinar a sequência de produção ideal para uma variedade de itens distintos representa um gargalo de produtividade para as empresas que operam em um regime HMLV, tais como produção de semicondutores, eletrônicos, peças usinadas e indústria aeroespacial (Gan; Musa; Yap, 2023).

A fim de lidar com uma grande variedade de produtos e requisitos complexos dos clientes, as fábricas fazem uso do modelo produtivo denominado sistema de manufatura flexível, (ou *flexible manufacturing system*, FMS) (Calmels, 2019). Um FMS consiste em um conjunto de máquinas flexíveis em uma linha de produção. Cada máquina flexível possui um *magazine* que pode ser configurado pela instalação de um conjunto de *ferramentas*, cada uma delas projetada para uma função como lixamento, corte, entre outros. A capacidade do *magazine* para ferramentas instaladas em cada máquina é limitada e, na maior parte das aplicações reais, é muito menor que a quantidade total de ferramentas disponível para processar todas as tarefas da produção.

No contexto do problema abordado nesta monografia, uma *tarefa* é definida como uma ou mais operações para fabricação de um produto. Cada operação possui um tempo de processamento e um conjunto de ferramentas necessárias para ser processada. A produção é divida em *estágios* em que cada tipo de operação é processada sem interrupções. O conjunto de ferramentas presentes no *magazine* de uma máquina em um estágio pode conter mais do que somente as ferramentas

necessárias para processar a dada operação de uma tarefa, porém, caso alguma ferramenta necessária para o processamento não esteja presente no *magazine* da máquina, é preciso realizar uma troca de ferramentas entre estágios consecutivos da produção.

Assim, se faz necessário determinar a sequência ótima para o processamento das operações, a fim de minimizar as trocas de ferramentas, uma vez que trocas resultam em custos adicionais. Na literatura, o custo da troca é representado tanto por interrupções na produção ou por uma penalidade concreta no lucro após o processamento de um conjunto de tarefas (Calmels, 2019). Esse desafio é referido na literatura como *job sequencing and tool switching problem* (SSP). O SSP envolve determinar a sequência ideal de processamento das tarefas que minimize o número de trocas de ferramentas, além de especificar o plano de troca de ferramentas em si, ou seja, quais ferramentas serão carregadas no *magazine* da máquina flexível a cada estágio de produção.

Existem diversas versões para o SSP, desde sua versão mais básica (Tang; Denardo, 1988) que considera uma sequência de tarefas a serem processadas em apenas uma máquina, até outras versões variando número de máquinas, objetivos únicos e diversos, ferramentas de tamanhos homogêneos e heterogêneos, lista completa de tarefas conhecida *a priori* ou não, capacidade do *magazine* e desgaste das ferramentas (Calmels, 2019).

A presente monografia trata de uma versão específica do SSP que consiste em minimizar o custo da produção em máquinas paralelas idênticas. Tarefas são classificadas entre *prioritárias* e *não prioritárias*, e o custo é associado ao número de trocas de ferramentas e ao processamento ou não de tarefas prioritárias. Além disso, as tarefas podem ser *reentrantes*, o que significa que pode haver a necessidade processá-las em mais de um estágio para sua fabricação. Outra característica do problema é a existência de um *período não supervisionado* de produção, de maneira que trocas de ferramentas não podem ser realizadas sem acompanhamento humano, embora a produção possa continuar dado que não sejam necessárias tais trocas. Essa recente versão do problema foi abordada uma única vez na literatura, recentemente por (Dang et al., 2023).

A fim de tratar o problema descrito, será utilizado o método revenimento paralelo (ou *parallel tempering*, PT) o qual se assemelha ao processo de reaquecimento e resfriamento de materiais visando melhorias em suas propriedades físicas. O PT não é popular na área de otimização, ao contrário do que ocorre na área de simulações físico-químicas. Isto não se deve ao desempenho do método, porém a questões históricas e de implementação (Almeida, 2024). Vale ressaltar que implementações paralelas do PT, apesar do nome, não são usuais na literatura, apesar de promissoras. A presente monografia emprega uma implementação paralela do PT a fim de produzir soluções para o problema abordado.

1.1 Justificativa

As aplicações do SSP no âmbito industrial são diversas, sendo uma das mais notórias em metalúrgicas. Essa indústria possui um mercado de mais de 200 bilhões de dólares, uma vez que esse tipo de manipulação de metal é usada em diversas outras indústrias, como aeroespacial e fabricação de automóveis, por exemplo (Chcompany, 2024). Nesse contexto, tornos ou máquinas de corte seriam as máquinas do problema e as ferramentas são acopladas as mesmas para processar as operações das tarefas. Dessa forma, abordar o SSP se mostra extremamente vantajoso para minimizar desperdícios e custos.

Outra aplicação do SSP ocorre na montagem de *printed circuit boards* (PCBs) (Ghrayeb; Finch, 2003), nesse problema, máquinas são alimentadas com componentes (análogos às ferramentas) a serem montados nas placas dos circuitos. Cada placa possui uma configuração específica de componentes, logo pode ser considerada uma tarefa. A indústria de montagem de PCBs tem grande relevância no mercado mundial e um mercado previsto em 73,1 bilhões de dólares até 2027 (IndustryARC, 2024).

Questões como reentrância e períodos de produção não supervisionada também são desafios relevantes dentro da abordagem do SSP, especialmente em ambientes de produção industrial. A reentrância ocorre quando uma mesma tarefa ou peça precisa retornar a uma máquina ou estágio anterior para retrabalho ou ajustes, o que dificulta o sequenciamento ótimo das operações e pode aumentar os custos e o tempo de produção. Já a produção não supervisionada envolve o funcionamento de máquinas por longos períodos sem intervenção humana direta, exigindo soluções que atinjam o máximo de produtividade de forma autônoma. A ausência de supervisão torna ainda mais crítica a eficiência das soluções, já que atrasos ou erros não podem ser corrigidos rapidamente, impactando diretamente na qualidade e no fluxo de produção.

Uma vez mostrada a importância do SSP para indústria, é importante ressaltar que o problema em questão pertence à classe NP-Difícil (Crama et al., 1994). Isto implica em, à medida que o tamanho do problema aumenta, encontrar uma solução ótima torna-se cada vez mais impraticável em um prazo razoável. Portanto, explorar abordagens alternativas para lidar com o SSP é crucial para o desenvolvimento de algoritmos eficientes que possam fornecer soluções satisfatórias dentro de restrições práticas.

O algoritmo revenimento paralelo foi recentemente explorado na literatura em problemas de sequenciamento de tarefas, mostrando-se eficaz ao alcançar bons resultados em termos de qualidade das soluções e eficiência computacional (Almeida; de Castro Lima; Carvalho, 2025b). Sua capacidade de equilibrar a exploração e a explotação no espaço de soluções, aliada à utilização de múltiplas cadeias de Markov em diferentes temperaturas, permite escapar de mínimos locais, tornando-o uma técnica promissora para o problema abordado neste trabalho.

1.2 Objetivos

Em linhas gerais, este trabalho visa empregar o método *revenimento paralelo* para abordagem da versão do SSP abordada com o intuito de avançar o estado da arte atual (Dang et al., 2023). Os objetivos específicos são:

1. Realizar uma revisão bibliográfica abrangente sobre o SSP, com foco na análise das características dos problemas similares ao tratado na presente monografia;
2. Gerar pesquisa e embasamento teórico a respeito SSP e suas variações;
3. Realizar pesquisa para geração de embasamento teórico a respeito do método *parallel tempering*;
4. Gerar instâncias desafiadoras para o problema a partir de dados reais, disponíveis publicamente por outros estudos;
5. Adaptar a API do PT para a versão específica do SSP (Almeida; de Castro Lima; Carvalho, 2025b);
6. Realizar experimentos preliminares para calibração de parâmetros;
7. Realizar experimentos computacionais;
8. Análise dos experimentos;

1.3 Organização do trabalho

A organização do trabalho segue a seguinte estrutura: o Capítulo 2 é dedicado a revisão dos trabalhos relacionados. O Capítulo 3 trata da fundamentação teórica tanto da versão do SSP em questão quanto do método a ser utilizado para sua abordagem. Em seguida, o Capítulo 4 aborda a metodologia empregada e o seu desenvolvimento, incluindo o detalhamento do método escolhido. O Capítulo 5 relata os experimentos computacionais e os resultados obtidos considerando instâncias baseadas em dados reais da literatura. Por fim, a conclusão deste estudo é apresentada no Capítulo 6.

2 Trabalhos relacionados

O SSP uniforme serve como base para vários problemas derivados, incluindo o tema desta monografia. O SSP uniforme destaca-se como o problema prevalente na pesquisa sobre problemas de otimização em sistemas de manufatura flexível. Originalmente introduzido por Tang e Denardo (1988), este problema específico envolve determinar a sequência ideal do processamento das tarefas e o plano de trocas de ferramentas correspondente com o objetivo de minimizar o número de trocas de ferramentas. A versão original do SSP é referida como "uniforme" devido aos tempos de configuração constantes e independentes da sequência, ao sequenciamento das tarefas em uma única máquina e aos tamanhos uniformes das ferramentas. O Capítulo 3 apresenta uma descrição detalhada deste problema.

No estudo de Tang e Denardo (1988) é realizada a conjectura de que o SSP é um problema pertencente à classe NP-Difícil, considerando que ele pode ser modelado como um problema do caixeiro viajante com arestas de comprimentos variados. Entretanto, não é apresentada uma prova formal para tal conjectura. Algum tempo depois, o estudo de Crama et al. (1994) prova formalmente que o SSP pertence de fato à classe de problemas NP-Difícil.

Atualmente, para o SSP uniforme, o estado da arte é definido por Mecler, Subramanian e Vidal (2020). Uma metaheurística denominada busca genética híbrida (ou *hybrid genetic search*, HGS) é aplicada à solução do SSP. O mesmo método já havia obtido bons resultados quando aplicado ao problema de roteamento de veículos. O estudo em questão gerou resultados melhores que abordagens anteriores por meio da utilização de um objetivo secundário, além da aplicação inédita do método HGS.

O estudo de Calmels (2019) oferece uma revisão ampla sobre o SSP e suas variantes. Foram analisados estudos considerando um período abrangente, desde 1988 até 2018, considerando cerca de 61 artigos no total. A análise é segmentada conforme as diferentes versões do problema, variando diversos parâmetros conforme a aplicação industrial, tais como os elencados a seguir.

Número de máquinas: Ambientes produtivos que lidam com uma máquina, máquinas paralelas idênticas (BARD, 1988; Beezão et al., 2017) e máquinas paralelas não relacionadas (Özpeynirci; Gökgür; Hnich, 2016).

Tempo de configuração do *magazine*: O tempo de configuração é constante para todas as ferramentas (BARD, 1988), ou dependente de outras características das ferramentas, como tamanho (Privault; Finke, 1995).

Objetivos: Minimizar o número de trocas de ferramentas (Crama et al., 1994) ou uma combinação de objetivos envolvendo a redução do tempo total de conclusão das tarefas e a minimização dos instantes de troca de ferramentas (Keung; Ip; Lee, 2001).

Tamanho das ferramentas e capacidade do *magazine*: Ferramentas que necessitam de mais de um espaço no *magazine* (Rupe; Kuo, 1997) e/ou tarefas que necessitam de mais ferramentas do que o *magazine* suporta simultaneamente (Matzliach; Tzur, 2000; Tzur; Altman, 2004).

Desgaste de ferramentas: Ferramentas que não se desgastam (BARD, 1988) ou desgaste de ferramentas representado por distribuição determinística ou estocástica (Hirvikorpi et al., 2006; Farughi et al., 2017).

A versão específica do SSP a ser abordada na presente monografia foi tratada em apenas um artigo, proposto por Dang et al. (2023), salvo melhor juízo. O problema se caracteriza por uma junção de diversas variações do SSP em um único problema adicionados de novas características. O mesmo se caracteriza pela utilização de múltiplas máquinas para processamento das tarefas, períodos de produção não supervisionada em que tarefas podem ser processadas, mas ferramentas não podem ser retiradas ou adicionadas no *magazine*, distinção entre tarefas prioritárias e não prioritárias, e, por fim, tarefas podem ter reentrância, exigindo que sejam processadas mais de uma vez para serem concluídas. O objetivo inédito desta versão do problema é maximizar o lucro, composto pela diferença entre o prejuízo, definido pela quantidade de tarefas prioritárias não concluídas e o custo das trocas de ferramentas, e o bônus gerado pela conclusão das tarefas regulares e prioritárias dentro do horizonte de planejamento.

A característica de períodos não supervisionados já foi abordada na literatura por Noël, Sodhi e Lamond (2007), visando selecionar a melhor velocidade de corte para o processamento de tarefas em máquinas flexíveis durante um período não supervisionado, porém, os aspectos de trocas de ferramentas não são considerados. Agnetis et al. (2009) abordam um período não supervisionado com probabilidade de falhas de tarefas durante esse período, impedindo que tarefas subsequentes sejam finalizadas. Contudo, o artigo não aborda trocas de ferramentas. As demais características, como tarefas reentrantes, tarefas prioritárias e horizonte de planejamento limitado não são encontrados em estudos relacionados a manufatura flexível.

O estudo de Dang et al. (2023) propõe três abordagens para o problema considerado nesta monografia: um modelo de programação linear inteira mista (ou *mixed-integer linear programming*, MILP), uma heurística de particionamento e um algoritmo genético. Conforme ocorre comumente em modelos de programação linear, apesar da garantia de otimalidade das soluções, o crescimento rápido do tempo de execução à medida que o tamanho das instâncias aumenta limita sua aplicação. Dessa maneira, a heurística de particionamento e o algoritmo genético obtiveram desempenho melhor para instâncias contendo mais do que 25 tarefas.

A pesquisa foi realizada com parceria de uma indústria real. Dessa maneira, as instâncias utilizadas para realização dos experimentos computacionais foram baseadas em dados reais, os quais foram disponibilizados publicamente pelos autores. Contudo, os dados disponibilizados não são suficientes para reproduzir as instâncias utilizadas no artigo, que não foram disponibilizadas.

Desta forma, é necessário realizar algumas suposições sobre as mesmas para tentar oferecer algum tipo de reprodutibilidade e comparabilidade. Além da criação de instâncias inéditas, também foi proposta a adaptação das instâncias disponibilizadas por Beezão et al. (2017) por meio da adição de informações como período não supervisionado, tarefas prioritárias e horizonte de planejamento.

O estudo realizado por Dang et al. (2023) estabelece o estado da arte para a versão do SSP em questão, porém, não é disponibilizado o código-fonte para nenhum dos três métodos utilizados no estudo. Os dados das tarefas disponibilizados pelos autores não contêm informação sobre quais tarefas são prioritárias ou não, sendo essa informação gerada dinamicamente para cada instância proposta. O mesmo ocorre com as informações sobre reentrância de cada tarefa. Esse fato, justamente com a não disponibilização das instâncias, dificulta a reprodutibilidade dos experimentos.

Uma análise preliminar dos dados disponibilizados indicou potenciais falhas, pois os mesmos contêm conjuntos de ferramentas maiores que a capacidade predefinida de 80 *slots* do *magazine*, mencionada no artigo, além de conjuntos de ferramentas que são subconjuntos de outros – o que implicaria na não necessidade de trocas de ferramentas. É válido ressaltar que é possível que parte dos dados redundantes não tenha sido utilizada na criação das instâncias. No entanto, sem acesso às instâncias, não é possível constatar ou não essa suspeita.

Outro aspecto problemático é que, após analisar os resultados publicados no artigo, os três métodos utilizados pelos autores conseguiram obter o valor ótimo para a função objetivo do problema em 38,4% das instâncias. Para 61,5% das instâncias, os resultados indicam que não houve penalidades associadas a trocas de ferramentas ou a tarefas prioritárias não finalizadas. Portanto, há indícios de que parte das instâncias pode não incluir dados relevantes sobre trocas de ferramentas e horizonte de planejamento, reduzindo assim o desafio para se obter bons resultados.

Levando em consideração as potenciais falhas apresentadas anteriormente, tornou-se necessária a criação de novas instâncias disponíveis publicamente para estudos futuros. Esta tarefa é descrita no Capítulo 5. As novas instâncias são baseadas nos dados reais disponibilizados por Dang et al. (2023), porém, evitando-se as deficiências identificadas nesta revisão da literatura.

3 Fundamentação Teórica

Nesse capitulo são formalizados os conceitos do SSP, da sua versão mais simples até sua versão específica tratada nesta monografia. Adicionalmente, é apresentada a base conceitual do método de resolução, o revenimento paralelo, com descrição de suas etapas de funcionamento.

3.1 O problema de minimização de troca de ferramentas

O SSP foi primeiramente abordado por Tang e Denardo (1988). Em sua versão uniforme, o problema pode ser descrito seguindo as seguintes especificações, de acordo com Calmels (2019):

- O conjunto de tarefas deve ser processado em uma única máquina;
- Os espaços reservados para cada ferramenta individualmente (*slots*) no *magazine* da máquina são idênticos e cada ferramenta requer apenas um espaço;
- Apenas uma ferramenta pode ser trocada por vez e os tempos de troca de ferramentas são constantes e iguais para todas as ferramentas;
- O conjunto de tarefas e o subconjunto de ferramentas necessárias para cada tarefa são conhecidos antecipadamente;
- O número de ferramentas exigido por qualquer tarefa é menor ou igual à capacidade do *magazine* de ferramentas; e
- As ferramentas não quebram nem se desgastam.

Dessa forma, o SSP pode ser dividido em dois subproblemas. O primeiro deles consiste em determinar uma permutação das tarefas na qual seja minimizada a quantidade de trocas de ferramentas no *magazine* da máquina. Esse subproblema pertence à classe NP-Difícil. O segundo subproblema consiste em, dada uma permutação das tarefas, determinar o plano de trocas de ferramentas, ou seja, determinar quais ferramentas estarão no *magazine* da máquina em cada estágio da produção. Esse problema pode ser resolvido em tempo determinístico polinomial pelo algoritmo *keep tool neeeded soonest* (KTNS) (Tang; Denardo, 1988) ou *greedy pipe construction algorithm* (GPCA) (Qiu et al., 2025).

Formalmente, o SSP uniforme pode ser descrito da seguinte maneira: dada uma máquina flexível com capacidade para armazenar C ferramentas em seu *magazine*, um conjunto de tarefas $T = \{1, \ldots, n\}$, um conjunto de ferramentas $F = \{1, \ldots, m\}$, o subconjunto de ferramentas

F_i $(F_i \in F)$ necessário para processar a tarefa i $(i \in T)$, o objetivo é determinar uma permutação dos elementos de T no qual o número de trocas de ferramentas seja minimizado. Na presente monografia, o problema abordado expande o SSP original com as seguintes especificidades, determinadas por Dang et al. (2023):

Múltiplas máquinas: As tarefas podem ser processadas por qualquer uma das múltiplas máquinas idênticas e cada máquina possui uma cópia própria do conjunto de ferramentas.

Horizonte de planejamento: Representa o prazo total para o processamento de tarefas. É considerado um horizonte de planejamento de 7 dias. Tarefas não finalizadas dentro do horizonte do plano podem acarretar em penalidades.

Período de produção não supervisionado: É determinado um período na produção em que as máquinas operam sem supervisão humana. Trocas de ferramentas não são permitidas nesse período, assim, todas as ferramentas requeridas devem estar presentes no *magazine* antes do início do período não supervisionado. É relevante evidenciar que o período não supervisionado ocorre sempre no fim do dia, ou seja, considerar um período não supervisionado de 12 horas significa reservar as 12 últimas horas do dia como não supervisionadas.

Trocas de ferramentas: As trocas não causam atraso na produção, mas possuem um custo associado. Quando ocorrem trocas de ferramentas em um determinado estágio da produção, elas pertencem a uma chamada *instância* de troca de ferramentas. Para cada instância de trocas existe um custo fixo e um custo variável. O custo fixo é associado a ocorrência de uma instância, e o custo variável é associado ao número de ferramentas trocadas em determinada instância. De acordo com Dang et al. (2023), o valor ideal para o custo fixo e variável são \$1 e \$10, respectivamente. É importante ressaltar que uma troca de ferramentas só acontece quando uma ferramenta necessária para uma operação não está presente na mesma máquina que processa a operação anterior. Dessa forma o carregamento inicial de ferramentas na máquina não representa uma troca.

Tarefas prioritárias: Cada tarefa pode ser *prioritária* ou *regular*. As tarefas prioritárias possuem um custo associado caso não sejam finalizadas dentro do horizonte de planejamento; para cada tarefa prioritária ou regular finalizada é atribuído um prêmio. De acordo com Dang et al. (2023), o prêmio por finalizar tarefas e o custo por não finalizar uma tarefa prioritária é de \$30.

Tarefas reentrantes: Tarefas podem possuir reentrância, ou seja, podem ter que ser processada mais de uma vez para serem finalizadas, i.e., possuem mais do que uma operação. Em termos práticos, significa que a tarefa é essencialmente duplicada, com o mesmo rótulo de prioridade e conjunto de ferramentas. Entretanto, o tempo de processamento pode ser diferente a cada operação. De acordo com Dang et al. (2023), tarefas podem ter reentrância de no máximo uma vez, o que corresponde a duas operações. Outra característica do

problema tratado por Dang et al. (2023) é a obrigatoriedade de se processar todas as operações de uma mesma tarefa em uma mesma máquina e a não obrigatoriedade de se processar todas as operações de uma tarefa.

Este é um problema NP-difícil por se tratar de um caso especial do SSP, o qual já é reconhecido como NP-difícil (Crama et al., 1994). O objetivo desta variante do SSP é a maximização do lucro, ou seja, a diferença entre o prêmio obtido pelo processamento das tarefas prioritárias, o custo das trocas de ferramentas (fixo mais variável) e a penalidade das tarefas prioritárias não finalizadas. A Figura 3.1 apresenta a relação entre prêmio e penalidade pela conclusão ou não das tarefas, assim como definido por Dang et al. (2023).

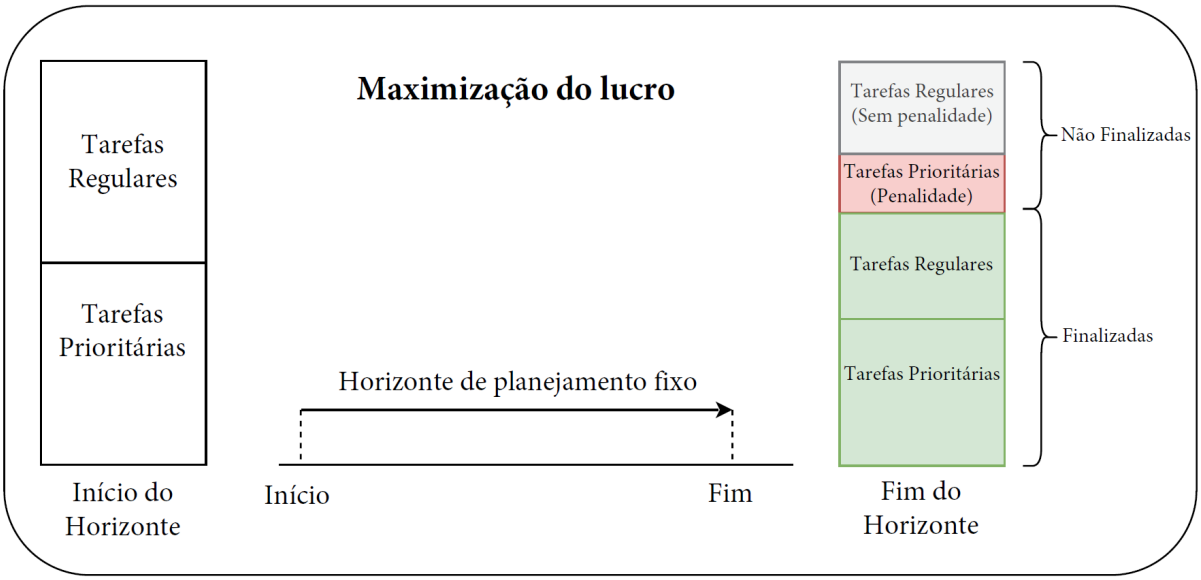

Figura 3.1 – Maximização do lucro com penalidades. Adaptado de Dang et al. (2023)

A Tabela 3.1 apresenta um exemplo de uma instância para o problema abordado. Tarefas possuem tempo de processamento, conjunto de ferramentas e rótulo de prioridade ou regularidade. Uma tarefa pode possuir uma ou duas operações, o que representa a reentrância. O conjunto de ferramentas faz referência a Tabela 3.2, na qual há um identificador para cada conjunto de ferramentas e a composição do conjunto de ferramentas em si, indicada pelos índices das ferramentas. No problema abordado, existe exatamente um exemplar de cada ferramenta por máquina, ou seja, mesmo que dois conjuntos contenham a mesma ferramenta, há apenas uma cópia disponível em cada máquina.

Tabela 3.1 – Exemplo de instância do SSP.

Tarefa	Operação	Tempo de Processamento	Conjunto de Ferramentas	Prioridade
1	1	3	F_1	Prioritário
1	2	5	F_1	Prioritário
2	1	7	F_2	Regular
3	1	6	F_3	Regular
3	2	8	F_3	Regular
4	1	4	F_2	Prioritário
4	2	9	F_2	Prioritário
5	1	6	F_4	Regular
6	1	10	F_5	Regular
7	1	5	F_1	Prioritário

Na instância em questão, o conjunto de todas as ferramentas é dado por $F = \{1, \ldots, 20\}$. É importante ressaltar a interseção entre os conjunto de ferramentas: os conjuntos F_2 e F_5, por exemplo, possuem o seguinte conjunto de ferramentas em comum: $F_2 \cap F_5 = \{15, 16, 17, 18\}$. Esse fato indica um potencial benefício em sequenciar estas tarefas próximas uma das outras a fim de diminuir o número de trocas de ferramentas.

Tabela 3.2 – Conjuntos de ferramentas para exemplo de instância SSP.

Conjunto de Ferramentas	Ferramentas	Tamanho
F_1	{1,2,3,4,5}	5
F_2	{12, 13, 14, 15, 16, 17, 18}	7
F_3	{4, 5, 8, 9, 10, 11, 12, 13}	8
F_4	{5, 6, 7}	3
F_5	{15, 16, 17, 18, 19, 20}	6

A Figura 3.2 representa o gráfico de Gantt de uma possível solução para o exemplo considerando um tempo não supervisionado de 12 horas (720 minutos) e um horizonte de planejamento de 2 dias (2880 minutos). São consideradas duas máquinas na produção e a capacidade do *magazine* para ambas é dada pelo tamanho do maior conjunto de ferramentas, ou seja, 8 ferramentas. A fim de identificar as tarefas, considera-se um par ordenado (*id*, *op*) em que *id* identifica a tarefa e *op* identifica a operação. A permutação $S_1 = [(1,1), (1,2), (2,1), (3,1), (3,2)]$ determina a sequência das tarefas na máquina 1 e a permutação $S_2 = [(4,1), (4,2), (5,1), (6,1)]$ determina a sequência de tarefas na máquina 2.

A Tabela 3.3 apresenta as ferramentas no *magazine* de cada máquina durante cada estágio da produção, bem como o número de trocas de ferramentas. É importante ressaltar que a inserção das ferramentas do primeiro estágio não são contadas como trocas. Considerando um custo de \$1 para o custo variável e \$10 para o custo fixo, o custo total das trocas de ferramentas para o

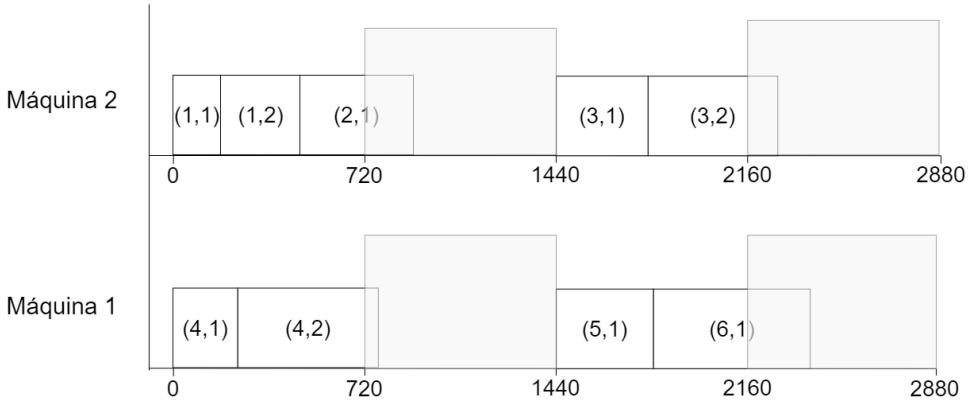

Figura 3.2 – Gráfico de Gantt para a primeira solução do exemplo.

processamento das tarefas na máquina 1 é dado por $(4 \times 1 + 10) + (5 \times 1 + 10) = 29$. O mesmo pode ser feito para a máquina 2: $(2 \times 1 + 10) + (2 \times 1 + 10) = 24$.

Tabela 3.3 – Sequenciamento das ferramentas para a primeira solução de exemplo.

	Estágio de Produção	Ferramentas no *magazine*	Número de Trocas
Máquina 1	(1, 1)	{1, 2, 3, 4, 5, 12, 13, 14}	-
	(1, 2)	{1, 2, 3, 4, 5, 12, 13, 14}	0
	(2, 1)	{4, 12, 13, 14, 15, 16, 17, 18}	4
	(3, 1)	{4, 5, 8, 9, 10, 11, 12, 13}	5
	(3, 2)	{4, 5, 8, 9, 10, 11, 12, 13}	0
Máquina 2	(4, 1)	{5, 12, 13, 14, 15, 16, 17, 18}	-
	(4, 2)	{5, 12, 13, 14, 15, 16, 17, 18}	0
	(5, 1)	{5, 6, 7, 12, 15, 16, 17, 18}	2
	(6, 1)	{5, 6, 15, 16, 17, 18, 19, 20}	2

Dado que 9 tarefas foram finalizadas dentro do horizonte de planejamento e que a tarefa (7,1) é prioritária e não foi finalizada, considerando um prêmio de \$30 por finalizar tarefas e um custo também de \$30 por não finalizar tarefas prioritárias, o valor da solução proposta é dado por $(9 \times\ 30) - (30 \times\ 1) - (29) - (24) = 187$.

A Figura 3.3 apresenta o gráfico de Gantt para uma segunda solução da instância de exemplo. É possível observar que, alternando a ordem das tarefas, foi possível processá-las todas dentro do horizonte de planejamento.

A Tabela 3.4 exibe a quantidade de trocas de ferramentas para cada máquina na segunda solução. Observa-se que houve um total de 10 trocas em 3 ocasiões distintas, resultando em um custo total de $\$40$. Nesta abordagem, todas as tarefas foram concluídas, resultando em um prêmio de $\$300$. Portanto, o valor da nova solução é de $\$260$, representando uma melhoria de cerca 40% em relação à solução anterior.

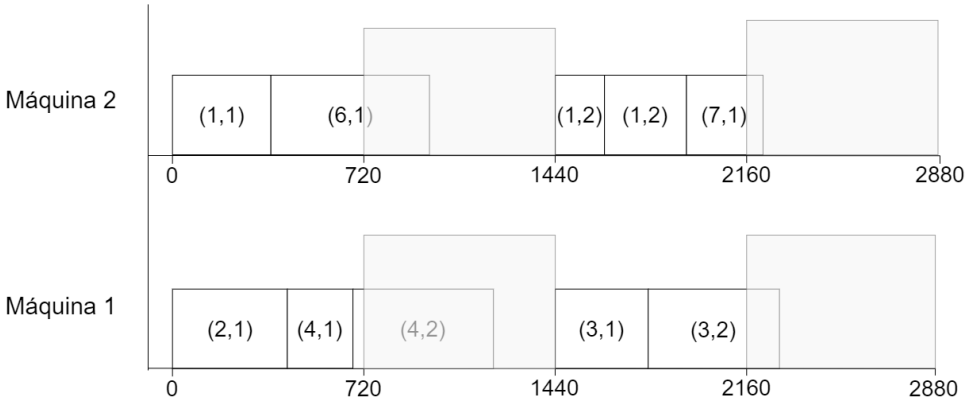

Figura 3.3 – Gráfico de Gantt para a segunda solução do exemplo.

Tabela 3.4 – Sequenciamento das ferramentas para a segunda solução do exemplo.

	Estágio de Produção	Ferramentas no *magazine*	Número de Trocas
Máquina 1	(2, 1)	{4,12,13,14,15,16,17,18}	-
	(4, 1)	{4,12,13,14,15,16,17,18}	0
	(4, 2)	{4,12,13,14,15,16,17,18}	0
	(3, 1)	{4,5,8,9,10,11,12,13}	5
	(3, 2)	{4,5,8,9,10,11,12,13}	0
Máquina 2	(5, 1)	{5,6,7,15,16,17,18,19}	-
	(6, 1)	{5,6,15,16,17,18,19,20}	1
	(1, 1)	{1,2,3,4,5,6,15,16}	4
	(1, 2)	{1,2,3,4,5,6,15,16}	0
	(7, 1)	{1,2,3,4,5,6,15,16}	0

3.2 Revenimento paralelo

O PT, também referido na literatura como *replica exchange*, reproduz o processo de revenimento, oriundo da metalurgia. Este é um método amplamente utilizado nas áreas de simulação em físico-química (Earl; Deem, 2005) e biologia (Hansmann, 1997). Entretanto, poucos trabalhos utilizaram o PT com aplicação em otimização.

Na ciência de materiais e metalurgia, revenimento é o processo de aquecer um metal até um certa temperatura por certo tempo e deixá-lo resfriar naturalmente no ambiente. Reaquecer o material em baixas temperaturas alivia tensões internas no material e reduz a fragilidade. Reaquecer em temperaturas mais altas implica em reduzir a dureza em troca de mais elasticidade e plasticidade (Almeida, 2024). O método computacional revenimento paralelo estabelece uma analogia entre o processo metalúrgico e o processo de solução de um problema de otimização. No processo metalúrgico, há uma relação entre dureza e tenacidade, já no processo de solução de problemas de otimização, há uma relação entre a diversificação e intensificação da busca (Almeida, 2024).

A fim de compreender o PT, primeiro devem ser entendidos os processos de Monte Carlo e amostragem iterativa com cadeias de Markov. O termo Monte Carlo surgiu no laboratório de desenvolvimento e pesquisa de Los Alamos, durante a criação da primeira bomba atômica (Metropolis et al., 1953). Dada a natureza secreta do projeto, os cientistas Nicholas Metropolis e Stanislaw Ulam sugeriram o nome Monte Carlo em referência a um grande cassino onde um tio de Stanislaw costumava apostar. Antes de se cunhar o nome Monte Carlo, o método era conhecido apenas como amostragem estatística. O termo Monte Carlo se refere a algoritmos que fazem uso de amostragem aleatória para resolver problemas.

Para problemas de otimização, o processo de Monte Carlo se dá pela simulação de diversas soluções e, no caso de problemas de maximização, escolher a de valor máximo entre todas as soluções avaliadas. Esse processo tende a convergir para o ótimo quando a quantidade de soluções exploradas é suficientemente grande, ou seja, o espaço de busca foi suficientemente explorado. A fim de melhorar a velocidade e a convergência do método, foi desenvolvido o método de Monte Carlo com cadeias de Markov (ou *Markov Chain Monte Carlo*, MCMC). Nesse caso, as soluções não são mais geradas de forma independente, mas sim de forma que uma solução s seja transformada em outra solução s' por meio da aplicação de um movimento.

Uma cadeia de Markov é uma forma de modelar um sistema que se move entre diferentes estados com certa probabilidade. A ideia principal é que o próximo estado dependa apenas do estado atual. Isso torna as cadeias de Markov úteis para estudar processos aleatórios que se alteram ao longo do tempo. Uma cadeia de Markov pode ser visualizada como um grafo direcionado como ilustrado na Figura 3.4, em que os vértices representam os estados, os arcos representam as transições entre os estados e o peso dos arcos dizem respeito a probabilidade de transição entre os estados.

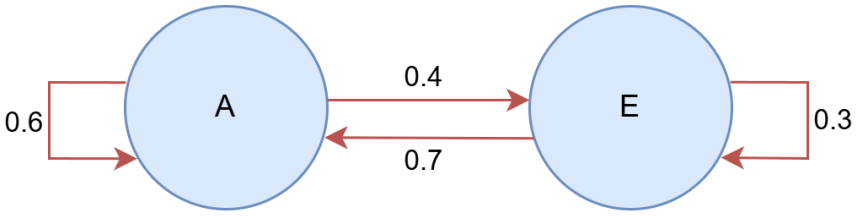

Figura 3.4 – Exemplo de uma cadeia de Markov.

A probabilidade do sistema se encontrar em um estado X_{n+1}, dado X_n como o estado anterior, é representada por $P\left(X_{n+1} = x \mid X_n = x_n\right)$. Tomando o exemplo da Figura 3.4, para determinar a probabilidade de o sistema se encontrar no estado E, basta saber qual o estado anterior. Considerando o estado anterior como A, temos $P\left(X_{n+1} = E \mid X_n = A\right) = 0,4$. Caso o estado anterior seja o E, temos $P\left(X_{n+1} = E \mid X_n = E\right) = 0,3$. O MCMC utiliza as cadeias de Markov para gerar soluções dependentes entre si, a partir de pequenas modificações entre uma solução e outra.

O Algoritmo de Metropolis é baseado no MCMC e é diretamente utilizado pelo PT. O algoritmo de Metropolis se baseia em construir uma cadeia de Markov com determinado

comprimento, em que os estados são soluções e as transições representam mudanças de uma solução para outra. O resultado do algoritmo é a melhor solução encontrada. É utilizado o conceito de temperatura para determinar a aceitação ou não de uma solução. A temperatura é representada por uma constante T e é utilizada na Equação 3.1, conhecida como função de distribuição de Boltzmann

$$P\left(\Delta E, T\right) = e^{-\frac{\Delta E}{T}} \tag{3.1}$$

em que e corresponde à constante conhecida como número neperiano, T é a constante de temperatura definida previamente e ΔE diz respeito a degradação da solução e pode ser expandido para $f\left(s'\right) - f\left(s\right)$, em que f corresponde a uma função de avaliação, s é a solução atual e s' é a nova solução.

Quando a temperatura T é alta em relação ao ΔE a aceitação de uma solução com degradação é provável já que a constante T domina as outras variáveis. Já em temperaturas mais baixas, a aceitação de soluções piores tem menos chance de acontecer. A variação da temperatura é o que garante ao PT o balanceamento entre diversificação e intensificação da exploração do espaço de busca.

O funcionamento geral do algoritmo de Metropolis pode ser observado no Algoritmo 3.1. O algoritmo recebe como entrada a temperatura T a ser utilizada para calcular a aceitação de soluções e um inteiro N_{kmax} que determina o comprimento máximo da cadeia de Markov a ser gerada. Na linha 1 a variável s_0 é inicializada com a solução inicial, a qual pode gerada aleatoriamente ou seguindo algum algoritmo. O laço de repetição entre as linhas 1 e 10 gera a cadeia de Markov, de comprimento N_{kmax}. Dentro do laço, na linha 4, é construída uma nova solução s' a partir da solução atual s. Na linha 5, é calculada o ΔE, ou degradação da solução. Na linha 6 é verificada se a nova solução s' é aceita a partir da função de Boltzmann, a qual considera a degradação da solução e a temperatura recebida como entrada do algoritmo. Caso a solução s' seja aceita, ela é considerada a solução a atual para a próxima iteração (linhas 6 e 7). Caso contrário, a solução atual será considerada novamente (linhas 8 e 9). Ao fim do algoritmo, a última solução aceita é retornada.

Algoritmo 3.1: Algoritmo de Metropolis

Entrada: Temperatura T e o comprimento da cadeia de Markov N_{kmax}.
Saída: Solução atual.

```
1  s_0 ← solução inicial;
2  k ← 0;
3  enquanto k ≤ N_kmax faça
4      s' ← q(s_k, s');
5      ΔE ← f(s') − f(s_k);
6      se aceita s' então
7          s_{k+1} ← s';
8      senão
9          s_{k+1} ← s_k;
10     k ← k + 1;
```

O PT coordena diferentes constantes T_i do algoritmo de Metropolis. São construídas k réplicas de um sistema – no nosso caso, um problema de otimização combinatória – em que cada uma considera uma temperatura T_i e uma solução inicial associada. Cada réplica consiste em um MCMC independente das demais. Periodicamente, são propostas trocas de temperaturas entre réplicas de temperaturas adjacentes. A troca de temperatura entre duas réplicas ocorre de acordo com uma probabilidade definida pela equação

$$P\left(r_i \ \rightarrow \ r_j\right) = min\left[1, \ exp\left(\Delta\beta\Delta E\right)\right] \tag{3.2}$$

em que $\Delta\beta$ pode ser expandido para $\frac{1}{T_j} - \frac{1}{T_i}$, com T_i e T_j correspondentes às temperaturas de cada réplica i e j envolvidas na proposta de troca, ao passo que ΔE indica a diferença entre o valor de cada solução em cada réplica, i.e., $f\left(r_j\right) - f\left(r_i\right)$.

Se a troca for aceita, a réplica r_i passa a considerar o valor da temperatura T_j e a réplica r_j passa a considerar o valor da temperatura T_i. Caso contrário, as temperaturas continuam as mesmas para r_i e r_j. Essa característica faz com que o sistema explore o espaço de busca de forma eficiente, gerando o equilíbrio entre diversificação, associado a temperaturas altas, em que soluções com degradação são mais prováveis de serem aceitas, e intensificação, associadas a temperaturas mais baixas, em que soluções piores são aceitas mais raramente.

O funcionamento geral do PT é ilustrado no Algoritmo 2. Como entrada, são recebidos um vetor de temperaturas T, um vetor de soluções iniciais S, um inteiro qT_{emp} que indica o número de tentativas de trocas de temperatura e um inteiro N_{kmax} que determina o tamanho da cadeia de Markov a ser gerada pelo algoritmo de Metropolis. O laço de repetição principal, entre as linhas 3 e 23, é executado até que o critério de parada seja atingido, que pode incluir, entre outros, um número definido de propostas de troca ou um limite de tempo de execução. O laço de repetição interno, entre as linhas 3 e 22, itera sobre todas as réplicas executando o algoritmo de Metropolis por meio do laço de repetição entre as linhas 8 e 15. A aceitação de novas soluções é

realizada caso esta seja melhor ou de acordo com a Equação 3.1. Após executar o algoritmo de Metropolis, é realizada uma tentativa de troca de temperaturas entre duas réplicas de temperatura adjacentes seguindo a probabilidade definida pela Equação 3.2 (linhas 17 a 22). A saída do algoritmo é a última solução aceita para cada réplica, i.e., em cada uma das temperaturas.

Algoritmo 3.2: Algoritmo genérico do PT

Entrada: Vetor de temperaturas T, vetor de soluções S, quantidade de trocas de temperaturas qT_{emp} e comprimento da cadeia de Markov N_{kmax}.

Saída: Vetor de soluções S.

```
1  enquanto CriterioParadaAtingido faça
2      indice_replica ← 0;
3      enquanto indice_replica < tamanho(T) faça
4          s ← S[indice_replica];
5          t ← T[indice_replica];
6          t' ← T[indice_replica − 1];
7          k ← 0;
8          enquanto k ≤ N_kmax faça
9              s' ← vizinho(s);
10             ΔE ← f(s') − f(s);
11             se ΔE ≤ 0 então
12                 s ← s';
13             senão se aceita s' então
14                 s ← s';
15             k ← k + 1;
16         S[indice_replica] ← s;
17         se indice_replica > 0 então
18             Δβ ΔE ← (1/t' − 1/t) × (f(S[indice_replica − 1]) − f(S[indice_replica]))
19             se Δβ ΔE ≥ 0 então
20                 troca(S[indice_replica], S[indice_replica − 1]);
21             senão se aceita a troca então
22                 troca(S[indice_replica], S[indice_replica − 1]);
23         indice_replica ← indice_replica + 1;
```

A Figura 3.5 ilustra a execução do PT, em que cada réplica é disposta em um nível na horizontal e cada quadrado representa a execução do algoritmo de Metropolis a uma determinada temperatura fixa. As temperaturas são fixas, e as réplicas se movimentam por elas. As setas na horizontal indicam que uma réplica manteve sua temperatura anterior e as setas na diagonal indicam que duas réplicas adjacentes trocaram de temperatura. Cada coluna na referida figura indica uma proposta de trocas de temperaturas entre réplicas adjacentes, totalizando qT_{emp}

tentativas.

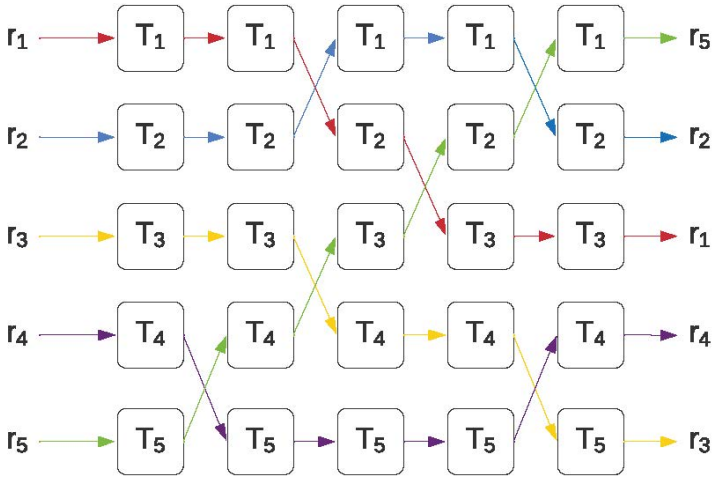

Figura 3.5 – Diagrama das trocas de temperaturas do algoritmo PT. Extraído de Almeida, de Castro Lima e Carvalho (2025b).

A implementação do PT a ser utilizada na presente monografia foi disponibilizada via API[1] por Almeida, de Castro Lima e Carvalho (2025b), o qual possui uma abordagem paralela para o PT. O paralelismo em CPU não é muito usual para implementações do PT, mas se mostrou promissor para resoluções de problema de otimização recentemente (Almeida; de Castro Lima; Carvalho, 2025b). Mesmo utilizando-se uma API, como a disponibilizada, ainda se faz necessário definir os componentes dependentes do problema e outros componentes complementares ao funcionamento do PT. Entre estes, citam-se:

Codificação: Estrutura de dados responsável por determinar a representação de uma solução.

Decodificação: Função responsável por transformar a solução codificada em uma solução completa para o problema tratado;

Soluções iniciais: Seguindo a codificação da solução, é necessário um método que retorne uma solução inicial para cada réplica do problema, seja de forma aleatória ou seguindo algum critério estruturado;

Avaliação da solução: Função responsável por receber uma solução e retornar um valor numérico que indique a qualidade da solução;

Geração de soluções vizinhas: Componente responsável por transformar uma solução s em uma solução s', para utilização pelas cadeias de Markov.

Critério de parada: Componente encarregado de determinar o fim da execução do método, podendo ser um definido como uma quantidade máxima de trocas de temperaturas ou um limite para o tempo de execução.

[1] Disponível em https://github.com/ALBA-CODES/PTAPI/. Acessado em 17 de setembro de 2024

Além de definir componentes e estruturas de dados para o problema em específico, também é necessário definir os parâmetros do PT via experimentos preliminares. Entre estes parâmetros, citam-se:

Temperatura Inicial: Valor da menor temperatura, que indica o início do intervalo de temperaturas;

Temperatura Final: Valor da maior temperatura, que indica o final do intervalo de temperaturas;

Replicas: Quantidade de réplicas a serem consideradas pelo algoritmo;

N_{kmax}**:** Comprimento das cadeias de Markov;

qT_{emp}**:** Quantidade de propostas de trocas entre temperaturas;

Método de inicialização da temperatura: Método da distribuição do intervalo de temperaturas entre as réplicas;

Método de atualização da temperatura: Método utilizado no ajuste dos valores das temperaturas de cada réplica;

Taxa de atualização da temperatura: Frequência de aplicação do ajuste das temperaturas, proporcional às propostas de troca.

Os próximos capítulos aprofundam-se no desenvolvimento da adaptação do PT básico para abordar o problema objeto de estudos desta monografia. Isso inclui a definição dos componentes requeridos para o funcionamento adequado do PT. Após a definição da implementação do PT, experimentos computacionais para avaliar a consistência dessa adaptação são reportados.

4 Desenvolvimento

Neste capítulo, é apresentada uma análise aprofundada dos dados disponíveis na literatura, bem como a descrição da criação de novas instâncias. Adicionalmente, serão descritas as características e adaptações realizadas na API utilizada.

4.1 Análise dos dados

Os dados utilizados na presente monografia foram fornecidos pela KMWE (*Klein Mechanishe Werkplaats Eindhoven*, ou *Small Mechanical Workshop in Eindhoven*), uma indústria de manufatura e engenharia de precisão dos Países Baixos. Esses dados foram disponibilizados por meio do trabalho de Dang et al. (2023) e são divididos em duas categorias principais: conjuntos de ferramentas e tarefas/máquinas. A categoria de conjuntos de ferramentas inclui uma lista de 3464 elementos, cada um contendo diversas ferramentas. As informações sobre tarefas/máquinas estão divididas em três outras coleções distintas:

- 2M376: contém 250 tarefas e 2 máquinas;
- 6M1201: contém 750 tarefas e 6 máquinas;
- 6M1401: contém 1000 tarefas e 6 máquinas.

Para cada tarefa nessas coleções, são fornecidas informações sobre o conjunto de ferramentas necessário e o tempo de processamento correspondente. No entanto, não há informações específicas sobre reentrância e prioridade de cada tarefa. Outros dados relacionados ao problema, descritos pelos autores durante o artigo, são como segue:

- Penalidade por tarefa prioritária não finalizada: \$30;
- Penalidade por instância de troca de ferramentas: \$10;
- Penalidade por troca de ferramenta: \$1;
- Bônus por tarefa prioritária finalizada: \$30;
- Horizonte de planejamento: 7 dias;
- Duração do período de não supervisão: 12 horas, na segunda metade de cada dia;
- Capacidade do *magazine*: 80 ferramentas.

A fim de analisar os dados e revelar possíveis falhas, uma primeira filtragem dos conjuntos de ferramentas consistiu em remover todos os conjuntos que não foram referenciados por nenhuma tarefa. Esse processo reduziu o número de conjuntos de ferramentas de 3464 para apenas 382. Desta forma, 3082 dos conjuntos de ferramentas utilizados na indústria parceira não foram utilizados nas instâncias geradas, o que representa 88,98% do total.

A versão do SSP em questão estabelece que todas as ferramentas para uma dada tarefa devem caber no *magazine* das máquinas, assim, conjuntos de ferramentas maiores do que a capacidade do *magazine* não devem ser permitidos. Apesar de haver a possibilidade de alguns conjuntos de ferramentas serem subconjuntos de outros, dada a natureza real dos dados, essa não é uma propriedade desejável. Instâncias com essa característica facilitam a solução do problema, pois basta agrupar as tarefas de tais conjuntos de ferramentas e sequenciá-los de maneira contígua, evitando trocas de ferramentas.

Dos conjuntos de ferramentas não referenciados, 30 possuem mais do que 80 ferramentas e 22 possuem 0 ferramentas. Dos 382 conjuntos de ferramentas referenciados, 3 possuem cardinalidade maior que 80 ferramentas. Dos 3082 conjuntos de ferramentas não referenciados, excluindo-se aqueles que são subconjuntos de outros e conjuntos contendo 0 ou mais que 80 ferramentas, sobram 1265 conjuntos de ferramentas únicos.

Ao analisar as coleções de tarefas individualmente, foram filtradas tarefas que utilizam conjuntos de ferramentas que são subconjuntos de outros na mesma coleção. Foi observado que diversas tarefas utilizavam exatamente o mesmo conjunto de ferramentas, o que também não é uma característica desejável. Essa filtragem resultou em 50 tarefas (2M376), 143 tarefas (6M1201) e 153 tarefas (6M1201), o que representa reduções de 80%, 81% e 84,7% nos conjuntos originais.

Conforme mencionado anteriormente no Capítulo 2, pela Tabela 5 de Dang et al. (2023), observa-se que, para diversas instâncias, é encontrado o valor ótimo para a função objetivo sem penalidade de troca de ferramentas. O valor ótimo para uma instância pode ser encontrado multiplicando o bônus de tarefas finalizadas pela quantidade de tarefas na instância. Esse valor indica que não houve nenhuma penalidade por troca de ferramentas nem por tarefa prioritária não finalizada. O mesmo fato ocorre para os diferentes métodos comparados nos experimentos computacionais do referido artigo, em diferentes tamanhos de instâncias. Esse fato indica que todas as tarefas puderam ser processadas dentro do horizonte de planejamento e não houve a necessidade de nenhuma troca de ferramenta, o que sugere a facilidade de resolver tais instâncias.

Essas inconsistências levaram o autor desta monografia a entrar em contato com os autores de Dang et al. (2023). O contato foi estabelecido e os autores mostraram boa vontade em esclarecer dúvidas e disponibilizar as instâncias utilizadas. No entanto, as respostas foram insuficientes para reproduzir as instâncias, uma vez que se fez uso de escolhas aleatórias para decidir quais tarefas seriam prioritárias e quais seriam reentrantes. As instâncias originais não foram enviadas, apesar da disposição inicial.

Dada a considerável redução na quantidade de dados utilizáveis e a falta de informação sobre prioridade e reentrância, tornou-se necessária a criação de novas instâncias. A próxima seção descreve este processo.

4.2 Novas instâncias

Novas instâncias foram criadas com base nos dados reais disponibilizados por Dang et al. (2023). A organização dos dados permanece a mesma para as novas instâncias, com a separação entre as tarefas e os conjuntos de ferramentas. Os dados referentes a penalidades, bônus, horizonte de planejamento, duração do período não supervisionado e capacidade do *magazine* permaneceram inalterados.

Foi criado um conjunto de 33 novas instâncias, baseadas nos 1265 conjuntos de ferramentas filtrados, posteriormente disponibilizado para uso da comunidade científica[1]. Desta forma, é garantido que as novas instâncias não terão as deficiências de conjuntos de ferramentas contidos em outros, conjuntos de ferramentas redundantes e nem conjuntos de ferramentas de cardinalidade igual a 0 ou maior que 80, citados anteriormente. Alem das instância, a implementação[2] e os resultados[3] completos então disponibilizadas na integra para o uso pela comunidade.

As instâncias geradas foram divididas em 11 dimensões, a partir de 75 tarefas e incrementando até 1236. Para cada dimensão, variou-se a porcentagem de tarefas prioritárias em 25%, 50% e 75%. Os nomes dos arquivos descrevem os detalhes da instância, no seguinte formato:

$$n = 75, p = 0.24, r = 0.5.csv \tag{4.1}$$

em que n representa o número de tarefas na instância, p a porcentagem de tarefas prioritárias, e r representa a porcentagem de tarefas reentrantes.

O número de tarefas geradas com base nos dados disponibilizados é maior do que o original. Desta forma, tornou-se necessário gerar novos tempos de processamento para as novas tarefas. Estes dados foram gerados de forma pseudoaleatória da seguinte maneira: foram recuperados e armazenados todos os tempos de processamento das coleções originais de tarefas. Em seguida, foi construída uma distribuição com esses valores, e os novos tempos de processamento foram gerados a partir desta distribuição. Essa abordagem garantiu que os novos valores não destoassem dos dados originais coletados por Dang et al. (2023).

Nestas novas instâncias, faz-se uma distinção clara entre tarefa e operação para indicar a reentrância. Uma tarefa pode conter mais de uma operação, indicando que essa tarefa possui reentrância. Como já estabelecido por Dang et al. (2023), tarefas podem conter no máximo uma

[1] https://github.com/mateus2k2/SSP/tree/master/input/Consolidated
[2] https://github.com/mateus2k2/SSP/
[3] https://github.com/mateus2k2/SSP/tree/master/output/TCC2V2

reentrada, ou seja, podem conter no máximo duas operações. Desta forma, o número de operações por tarefa é explicitamente mencionado na instância.

A porcentagem de tarefas reentrantes para cada instância foi determinada com base na quantidade total de tarefas. Em comunicação privada, os autores de Dang et al. (2023) informaram que as porcentagens de tarefas reentrantes variavam conforme o tamanho das instâncias: 50% para 376 tarefas, 60% para 1201 tarefas e 40% para 1401 tarefas. Portanto, para as novas instâncias, as mesmas porcentagens mantidas, atribuídas de acordo com a proximidade do número de tarefas das instâncias em relação aos dados da recomendação dos autores de Dang et al. (2023).

4.3 Pré-processamento

Nas novas instâncias criadas, é realizado um pré-processamento para agrupar as operações de uma mesma tarefa. Nesse processo, considera-se uma única tarefa cujo tempo de processamento corresponde à soma dos tempos individuais de ambas as operações. No entanto, ao avaliar a conclusão ou interrupção da tarefa, os valores são analisados separadamente para cada operação.

No trabalho de Dang et al. (2023), o mesmo agrupamento é realizado no algoritmo genético, já no modelo matemático, esse processo não é aplicado diretamente, sob a justificativa de proporcionar maior liberdade ao método. Além disso, os autores argumentam que prolongar a produção em períodos não supervisionados pode ser prejudicial à solução por ocasionar mais trocas de ferramentas no período supervisionado. Entretanto, discorda-se nesta monografia que essa abordagem seja a mais eficiente, pois o agrupamento reduz o número de tarefas a serem escalonadas, além de garantir melhor uso do período não supervisionado e garantir que as operações de uma mesma tarefa sejam realizadas na ordem correta.

Sem o agrupamento, soluções inviáveis podem surgir, como a execução de uma operação 2 antes da operação 1 de uma determinada tarefa reentrante. Portanto, considera-se nesta monografia que o agrupamento é a melhor estratégia para as instâncias propostas. Esse pré-processamento garante que todas as operações de uma mesma tarefa sejam alocadas em uma mesma máquina, como estabelecido por (Dang et al., 2023). Contudo, essa restrição não é totalmente justificada, uma vez que as máquinas paralelas são idênticas e não permitir o paralelismo do processamento das tarefas pode prejudicar a solução.

A Tabela 4.1 apresenta o resultado do pré-processamento da instância apresentada na Tabela 3.1. As tarefas 1, 3 e 4, tiveram suas operações reentrantes unificadas em uma única operação cada, na qual o tempo de processamento foi somado, enquanto a prioridade e o conjunto de ferramentas permanecem os mesmos.

Tabela 4.1 – Exemplo instância pré-processada SSP.

Tarefa	Operação	Tempo de Processamento	Conjunto de Ferramentas	Prioridade
1	1	8	F_1	Prioritário
2	1	7	F_2	Regular
3	1	14	F_3	Regular
4	1	13	F_2	Prioritário
5	1	6	F_4	Regular
6	1	10	F_5	Regular
7	1	5	F_1	Prioritário

Dessa forma, o número de tarefas é reduzido em 33%, o que resulta em uma diminuição da dificuldade em resolver o problema. Essa redução impacta diretamente o escalonamento das tarefas pelo método, tornando o processamento mais eficiente e permitindo uma melhor alocação dos recursos disponíveis. Além disso, a diminuição da quantidade de tarefas pode levar a um tempo de execução menor, otimizando o desempenho do sistema como um todo.

4.4 Codificação e decodificação

A codificação da solução é realizada por meio de um único arranjo com n itens, em que cada um deles corresponde ao índice de uma tarefa. Desta forma, trata-se o problema como se houvesse apenas uma máquina para processar todas as tarefas. A decodificação é realizada considerando um horizonte de planejamento m vezes maior que o original, em que m é o número de máquinas disponíveis para processar as tarefas. Dessa forma, o algoritmo KTNS é executado apenas uma vez, mesmo que haja mais de uma máquina. A divisão de tarefas entre as diferentes máquinas é realizada de maneira lógica, de acordo com os intervalos definidos pelo tamanho do horizonte de planejamento original.

A Figura 4.1 ilustra o processo de codificação e decodificação. Na parte superior, tem-se a solução codificada em um único arranjo. Na parte inferior, tem-se a solução decodificada considerando um horizonte de planejamento de 4 dias e considerando duas máquinas para o processamento das tarefas. Cada retângulo sólido representa uma tarefa sendo processada e é rotulado novamente como (*id*, *op*), em que *id* identifica a tarefa e *op* identifica a operação. Os retângulos opacos indicam os períodos de produção não supervisionada. O eixo x denota o tempo, em minutos, de 0 até 5760 (48 horas). A divisão das tarefas entre as máquinas é indicada pelas linhas verdes.

De acordo com a abordagem de decodificação, torna-se necessário criar um mecanismo que considere situações de inviabilidade. Por exemplo, uma tarefa que exija trocas de ferramentas durante o período não supervisionado, terá seu processamento postergado para o próximo período supervisionado, vide alocação da tarefa $(1, 1)$ no instante 1440. O mesmo ocorre caso o

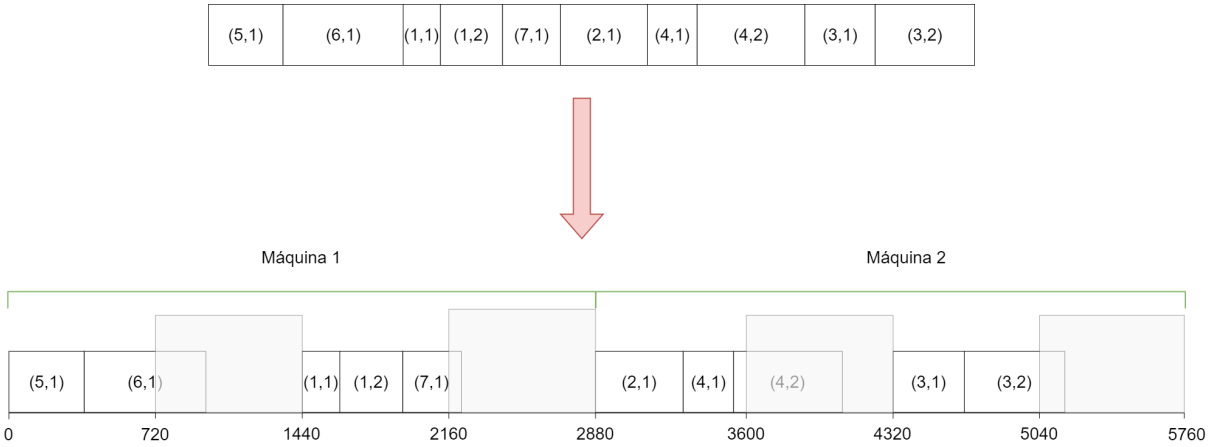

Figura 4.1 – Exemplo codificação e decodificação de uma solução.

processamento de uma tarefa exceda o horizonte de planejamento para uma máquina, pois isso implicaria no processamento da tarefa começar em uma máquina e terminar em outra. Esses problemas são mitigados através de contadores de tempo e verificações periódicas durante a execução do KTNS adaptado.

4.5 Solução inicial

A fase de solução inicial desempenha um papel crucial nas metaheurísticas, pois oferece uma base sobre a qual as melhorias subsequentes serão realizadas. A abordagem adotada para gerar soluções iniciais é criar uma permutação de todas as tarefas e suas respectivas operações, conforme o tamanho da instância. Essas permutações são geradas de forma aleatória. Por exemplo, considere uma instância de 7 tarefas e 9 operações, como o exemplo da Tabela 3.1. Cada tarefa é representada por um par ordenado (*id*, *op*) em que *id* identifica a tarefa e *op* identifica a operação. Uma possível permutação é $[(1,1),(6,1),(5,1),(1,2),(7,1),(2,1),(4,1),(4,2),(3,1),(3,2)]$.

Cada permutação representa uma solução inicial distinta, que será posteriormente codificada e decodificada pelo PT. Esse processo permite que diferentes pontos de partida sejam explorados pela metaheurística, aumentando a diversidade das soluções e melhorando, potencialmente, a qualidade das soluções finais encontradas.

A fim de melhorar a qualidade da solução inicial, foi desenvolvida uma heurística. Primeiramente, as tarefas da entrada são separadas em dois grupos: prioritárias e não prioritárias. Em seguida, é gerada uma permutação para cada um dos grupos, que são posteriormente concatenados nesta mesma ordem. Tomando como exemplo a instância da Tabela 4.1, uma possível permutação das tarefas prioritárias é dada por

$$[(1,1),(7,1),(4,1)].$$

Analogamente, uma permutação das tarefas não prioritárias é dada por

$$[(6,1),(2,1),(5,1),(3,1)].$$

A solução inicial resultante é obtida pela concatenação das duas permutações, ou seja,

$$[(1,1),(7,1),(4,1),(6,1),(2,1),(5,1),(3,1)].$$

Como a não finalização de tarefas prioritárias resulta em um prejuízo, enquanto isso não ocorre com as tarefas regulares, é natural assumir que escalonar todas as tarefas prioritárias no início do processamento leva a uma melhoria mais rápida da solução. Como a mencionada heurística faz uso de aleatoriedade, é possível gerar soluções iniciais distintas para cada uma das réplicas do PT. É importante ressaltar que, caso haja alguma vantagem relacionada à troca de ferramentas que torne o escalonamento de uma tarefa prioritária no final mais vantajoso, o método de resolução tem liberdade para explorar essa possibilidade ao longo da sua execução.

4.6 Função de avaliação

Dada uma sequência de processamento de tarefas, a avaliação padrão consiste em determinar o plano ótimo para as trocas de ferramentas, o que pode ser feito pelo algoritmo KTNS (Tang; Denardo, 1988). Na prática, dada uma sequência de tarefas, o KTNS as avalia e determina a sequência em que as ferramentas serão necessárias. Mantendo uma lista de prioridades dinâmica, o algoritmo garante que as ferramentas requeridas sejam mantidas no *magazine*, enquanto aquelas não requeridas imediatamente são desconsideradas. Havendo a necessidade de trocas de trocas de ferramentas, são mantidas no *magazine* prioritariamente aquelas que serão utilizadas mais em breve. A implementação do KTNS utilizada na presente monografia foi adaptada da implementação sequencial de melhor desempenho encontrada na literatura, disponibilizada por Almeida, de Castro Lima e Carvalho (2025a).

O problema tratado nesta monografia considera um período de produção não supervisionado e um horizonte de planejamento para calcular o valor da função objetivo de uma dada sequência de tarefas. Caso o processamento de tarefas ultrapasse o horizonte de planejamento, apenas as tarefas finalizadas até o término deste horizonte devem ser contabilizadas no bônus, enquanto as outras resultarão em penalidades. Além disso, se uma tarefa iniciar seu processamento durante o período não supervisionado e requerer que alguma ferramenta seja adicionada ao *magazine*, o início do processamento deverá ser postergado para o início do próximo período supervisionado.

As peculiaridades do problema em questão fazem com que o KTNS precise ser adaptado para considerar as características adicionais. A princípio, todas as tarefas a serem processadas em uma dada máquina são enviadas para o algoritmo. À medida que o plano de ferramentas é construído, o número de trocas de ferramentas e a quantidade de instantes de troca são acumulados. Além disso, o tempo de processamento é acumulado em um contador de tempo. Isso permite realizar verificações de período não supervisionado e de horizonte de planejamento. O algoritmo termina quando todas as tarefas da solução são processadas ou quando o contador de tempo

alcança o horizonte de planejamento. O valor da solução é então retornado: bônus das tarefas finalizadas, menos penalidade por tarefas prioritárias não finalizadas e trocas de ferramentas.

4.7 Estruturas de vizinhança

A determinação de estruturas de vizinhança é um dos elementos fundamentais na busca local para problemas de sequenciamento de tarefas, permitindo explorar o espaço de soluções a partir de pequenas modificações em uma solução atual. Esta exploração visa gerar soluções similares, que podem melhorar a solução atual ou introduzir diversificação, possibilitando a fuga de ótimos locais e contribuindo para a eficácia de metaheurísticas e métodos semelhantes. As seguintes estruturas de vizinhança são consideradas:

2-swap: Dada duas posições definidas dentro da sequência de tarefas, troca-se os dois elementos correspondentes às estas duas posições. Especificamente na implementação utilizada, as posições são definidas aleatoriamente. Seguindo essa estrutura, a solução $[(1,1),(1,2),(2,1),(3,1),(3,2),(4,1),(4,2),(5,1),(6,1),(7,1)]$ se transforma em $[(1,1),(1,2),(2,1),(4,2),(3,2),(4,1),(3,1),(5,1),(6,1),(7,1)]$ com observado na Figura 4.3 após a definição das posições 3 e 6.

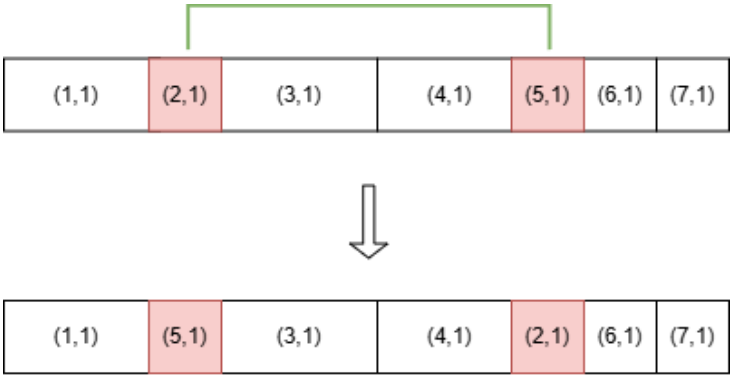

Figura 4.2 – Exemplo movimento 2-*swap*.

Inserção: Dada duas posições definidas dentro da sequência de tarefas, o elemento da primeira posição é inserido na segunda posição. Especificamente na implementação utilizada, as posições são definidas aleatoriamente. Como se observa no exemplo na Figura 4.3 a solução $[(1,1),(2,1),(3,1),(4,1),(5,1),(6,1),(7,1)]$ se transforma em $[(1,1),(3,1),(4,1),(5,1),(2,1),(6,1),(7,1)]$ após a definição das posições 2 e 5.

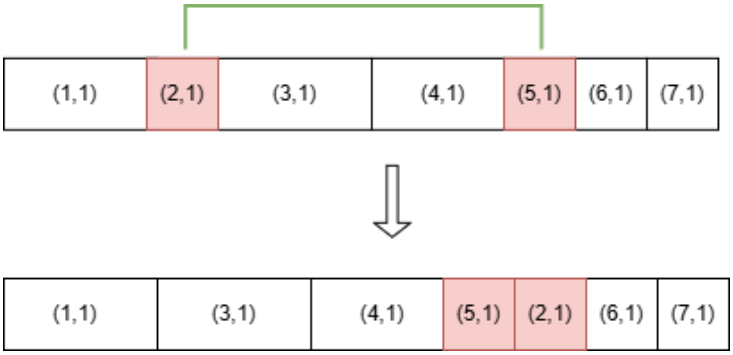

Figura 4.3 – Exemplo movimento inserção.

2-opt: Dada duas posições definidas dentro da sequência de tarefas, inverte-se a ordem de todos os elementos no intervalo definido pelas posições. Especificamente na implementação utilizada, as posições são definidas aleatoriamente. Como pode ser observado na Figura 4.4, a solução $[(1,1),(2,1),(3,1),(4,1),(5,1),(6,1),(7,1)]$ se transforma em $[(1,1),(5,1),(4,1),(3,1),(2,1),(6,1),(7,1)]$ após a definição das posições 2 e 5.

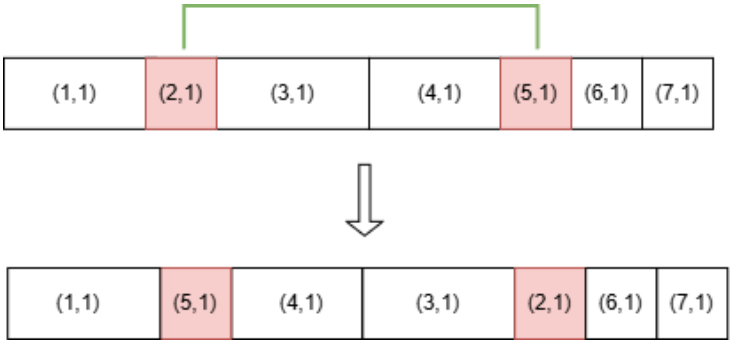

Figura 4.4 – Exemplo movimento 2-opt.

Com a implementação da função de avaliação, codificação e estrutura de vizinhança concluída, o próximo capítulo é dedicado à realização de experimentos preliminares.

5 Experimentos Computacionais

Neste capítulo, são apresentados os experimentos realizados com o intuito de validar a abordagem proposta para o problema em estudo. Além disso, foi realizada uma avaliação da consistência dos conjuntos de instâncias criados. Os experimentos computacionais foram realizados em um computador Intel Core i7-10700 CPU @ 2.90GHz com 8 cores e 16 GB de RAM sob o sistema operacional Ubuntu 21.10. O código foi implementado em C++, compilado com g++ 11.2.0 e opção de otimização -O3.

5.1 Instâncias

A Tabela 5.1 apresenta as características das instâncias geradas, como a quantidade de tarefas, a taxa de tarefas prioritárias, a taxa de tarefas reentrantes, o horizonte de planejamento, o limite inferior para o horizonte de planejamento (dado pela soma dos tempos de processamento dividido pelo número de máquinas) e o número de máquinas. Em todas as instâncias, o *magazine* pode armazenar até 80 ferramentas, i. e., $C = 80$, e nenhuma operação utiliza mais do que esta quantidade de ferramentas.

Tabela 5.1 – Características das instâncias.

Tarefas	Taxa de Prioritárias	Taxa de Reentrantes	Quantidade de Ferramentas	Horizonte de planejamento	Limite inferior para horizonte	Máquinas
75	{0,24; 0,51; 0,75}	0,5	650	2	{2,40; 2,54; 2,18}	2
212	{0,25; 0,50; 0,75}	0,4	1390	6	{6,94; 6,63; 6,79}	2
228	{0,25; 0,50; 0,75}	0,6	1520	6	{6,95; 6,86; 7,32}	2
399	{0,25; 0,50; 0,75}	0,5	2310	10	{12,4; 12,4; 12,2}	2
499	{0,25; 0,50; 0,75}	0,5	2690	4	{5,24; 5,33; 5,27}	6
600	{0,25; 0,50; 0,75}	0,5	3096	5	{6,40; 6,34; 6,37}	6
699	{0,25; 0,50; 0,75}	0,5	3420	6	{7,26; 7,39; 7,36}	6
800	{0,25; 0,50; 0,75}	0,6	3527	7	{8,23; 8,40; 8,52}	6
899	{0,25; 0,50; 0,75}	0,6	3746	8	{9,17; 9,27; 9,52}	6
1000	{0,25; 0,50; 0,75}	0,6	3950	9	{10,3; 10,3; 10,9}	6
1236	{0,25; 0,50; 0,75}	0,6	4431	11	{13,0; 13,3; 13,2}	6

5.2 Ajuste de parâmetros

O PT necessita que sejam definidos alguns parâmetros para a execução do algoritmo. Para os experimentos apresentados neste capítulo, esses parâmetros foram determinados usando a ferramenta *irace* (López-Ibáñez et al., 2016), um pacote que pode ser encontrado na linguagem R. O *irace* é um método *offline* de configuração automática de algoritmos de otimização. Dado um subconjunto das instâncias do problema para treino, o método determina a combinação de valores mais apropriada para os parâmetros para utilização geral. Os parâmetros usados para os

experimentos e seus respectivos valores são apresentados na Tabela 5.2. Os valores selecionados estão indicados em negrito.

Tabela 5.2 – Dados utilizados pelo *irace*. Os valores selecionados estão indicados em negrito.

Parâmetros	Opções
Temperatura inicial	$\{0,01;\ \mathbf{0,1};\ 0,2\}$
Temperatura final	$\{0,5;\ 1;\ \mathbf{5}\}$
N_{kmax}	$\{200; 300; 400; \mathbf{500}\}$
Distribuição inicial de temperatura[a]	$\{1; 2; \mathbf{3}; 4\}$
Tipo de movimento[b]	$\{\mathbf{1}; 2; 3\}$
Tipo de solução inicial[c]	$\{1; \mathbf{2}\}$
Tipo de ajuste da temperatura[d]	$\{0; \mathbf{1}; 2; 3\}$
Proporção para taxa de ajuste da temperatura	$\{\mathbf{3}; 4; 5\}$

[a] 1 - Linear, 2 - Linear inversa, 3 - Exponencial, 4 - Progressão geométrica.
[b] 1 - 2-opt, 2 - 2-swap, 3 - Inserção aleatória.
[c] 1 - Totalmente Aleatória, 2 - Aleatória com prioritários primeiro.
[d] 1 - 23%, 2 - Igualar taxas de aceitação, 3 - *Feedback-optimized*.

Foram utilizadas 11 quantidade de temperaturas, ou seja, 11 replicadas para a execução dos experimentos. Alem disso, é importante destacar que os valores de qT_{emp} e de N_{kmax} foram ajustados com base na convergência do método e em um tempo de execução viável, sendo definidos como 600 propostas de troca e tamanho de 500 para o comprimento da cadeia de markov. A taxa de ajuste da temperatura foi definida proporcionalmente ao valor de qT_{emp}, com uma proporção determinada pelo *irace* como 3. Isso significa que a taxa de ajuste da temperatura foi estabelecida como um terço do qT_{emp}, resultado em um ajuste a cada 200 prosposta de troca. Esse valor pode ser considerado razoável, pois um valor de 2 não seria efetivo, resultando em apenas duas atualizações da temperatura ao longo da execução.

5.3 Experimentos computacionais

Os resultados obtidos nos experimentos computacionais são apresentados na Tabela 5.3. Nessa tabela, o valor da solução objetivo foi decomposto entre o componente de bônus por tarefas finalizadas e os três componentes de penalidade. Esta é a primeira vez que esses dados são apresentados na literatura, uma vez que o trabalho de Dang et al. (2023) não disponibilizou suas instâncias nem o código para testes. Para cada instância, são apresentados o identificador, o número de tarefas finalizadas, o número de tarefas prioritárias não finalizadas, o número de tarefas não finalizadas, o número de instâncias de trocas de ferramentas e o número de trocas de ferramentas. Todos os valores são médios, calculados a partir de 10 execuções independentes do método por instância.

Tabela 5.3 – Componentes de bônus e penalidades da função objetivo.

Instância	Tarefas finalizadas	Tarefas prioritárias não finalizadas	Total tarefas não finalizadas	Instâncias de troca	Trocas de ferramentas
1-n=75,p=0,24,r=0,5,t=650	54,20	0,00	20,80	18,30	252,10
2-n=75,p=0,51,r=0,5,t=650	50,10	1,70	24,90	19,50	294,10
3-n=75,p=0,75,r=0,5,t=650	59,40	2,00	15,60	23,30	358,90
4-n=212,p=0,25,r=0,4,t=1390	153,70	1,80	58,30	62,70	628,70
5-n=212,p=0,50,r=0,4,t=1390	161,00	11,40	51,00	72,40	730,70
6-n=212,p=0,75,r=0,4,t=1390	155,60	30,40	56,40	68,50	622,60
7-n=228,p=0,25,r=0,6,t=1520	158,40	0,40	69,60	65,50	672,30
8-n=228,p=0,50,r=0,6,t=1520	159,50	5,50	68,50	66,40	845,70
9-n=228,p=0,75,r=0,6,t=1520	153,30	25,20	74,70	62,40	745,20
10-n=399,p=0,25,r=0,5,t=2310	257,10	1,90	141,90	110,70	934,00
11-n=399,p=0,50,r=0,5,t=2310	257,20	20,30	141,80	112,70	1015,20
12-n=399,p=0,75,r=0,5,t=2310	258,50	60,00	140,50	110,10	1075,80
13-n=499,p=0,25,r=0,5,t=2690	297,70	8,50	201,30	125,60	1093,60
14-n=499,p=0,50,r=0,5,t=2690	291,00	26,90	208,00	125,80	1080,80
15-n=499,p=0,75,r=0,5,t=2690	299,40	94,60	199,60	123,40	1147,70
16-n=600,p=0,25,r=0,5,t=3096	364,10	9,20	235,90	152,70	1305,90
17-n=600,p=0,50,r=0,5,t=3096	363,80	36,40	236,20	160,80	1405,50
18-n=600,p=0,75,r=0,5,t=3096	359,40	114,50	240,60	143,40	1414,30
19-n=699,p=0,25,r=0,5,t=3420	435,50	9,70	263,50	189,90	1616,10
20-n=699,p=0,50,r=0,5,t=3420	422,50	46,40	276,50	193,80	1499,20
21-n=699,p=0,75,r=0,5,t=3420	431,70	115,60	267,30	186,20	1733,40
21-n=800,p=0,25,r=0,6,t=3527	510,10	12,70	289,90	216,60	1815,50
22-n=800,p=0,50,r=0,6,t=3527	500,00	35,80	300,00	214,50	1876,40
23-n=800,p=0,75,r=0,6,t=3527	499,20	117,20	300,80	210,20	2142,30
24-n=899,p=0,25,r=0,6,t=3746	582,90	9,50	316,10	247,50	1966,50
25-n=899,p=0,50,r=0,6,t=3746	571,60	47,30	327,40	243,00	2223,50
26-n=899,p=0,75,r=0,6,t=3746	573,80	134,10	325,20	240,70	2347,60
27-n=1000,p=0,25,r=0,6,t=3950	643,00	12,30	357,00	276,10	2181,60
28-n=1000,p=0,50,r=0,6,t=3950	639,70	47,50	360,30	281,80	2609,60
29-n=1000,p=0,75,r=0,6,t=3950	618,60	165,70	381,40	265,30	2672,80
31-n=1236,p=0,25,r=0,6,t=4431	775,40	17,40	460,60	341,60	2562,70
32-n=1236,p=0,50,r=0,6,t=4431	741,90	73,70	494,10	329,60	2761,80
33-n=1236,p=0,75,r=0,6,t=4431	755,60	212,10	480,40	323,50	3073,60

Analisando os componentes da função objetivo, é possível observar uma estabilidade do PT em cada um dos tamanhos de instância. Primeiramente, ao examinar a quantidade de tarefas finalizadas, nota-se que ela se mantém relativamente constante para cada grupo. Dentre todas as execuções das instâncias 16, 17 e 18, obteve-se um máximo de 371 tarefas finalizadas e um mínimo de 354, o que indica uma estabilidade do PT em relação à porcentagem de tarefas prioritárias finalizadas nas instâncias. Logicamente, o mesmo fenômeno ocorre em relação ao total de tarefas não finalizadas. Isso era esperado, pois as instâncias possuem as mesmas tarefas com os mesmos tempos de processamento. Assim, embora a quantidade total de tarefas varie pouco, as tarefas escolhidas para serem finalizadas diferem entre as instâncias.

Observando as tarefas prioritárias não finalizadas na segunda coluna da Tabela 5.3, é possível concluir que tarefas prioritárias são de fato privilegiadas no escalonamento das tarefas pelo PT, havendo, em média, 6,85% de tarefas prioritárias não finalizadas, com no máximo 19% e mínimo de 0%. Apesar disso, identifica-se uma tendência clara de crescimento de não finalização dessas tarefas à medida que se aumenta a taxa de prioritárias na instância. Contudo, como a quantidade de tarefas finalizadas se mantém praticamente constante para instâncias de

mesmo tamanho, é intuitivo concluir que esse aumento ocorre simplesmente porque as tarefas prioritárias representam cada vez mais o total de tarefas na instância, dessa maneira, como o PT não consegue escalonar mais tarefas, as prioritárias também não são finalizadas.

A Tabela 5.4 apresenta o valor da solução inicial, S_0, o valor da melhor solução encontrada, definido por S^*, a solução média entre todas as execuções, expressa por S, o desvio padrão percentual, definido por $\sigma(\%)$, e o tempo médio de execução, representado, em segundos, pela coluna $T(s)$. A coluna de convergência expressa o percentual de qT_{emp} utilizado para obter a melhor solução. Todos os valores são médios, calculados a partir de 10 execuções independentes do método por instância. Por fim, o *gap*, ou distância percentual, é definido pela equação 5.1.

$$gap = \frac{S^* - S_0}{S^*} \times 100 \tag{5.1}$$

Tabela 5.4 – Resultados detalhados.

Instância	S_0	S^*	S	σ (%)	$T(s)$	Convergência (%)	*gap* (%)
1-n=75,p=0,24,r=0,5,t=650	645,10	1196,00	1190,90	0,22	57,71	66,33	46,06
2-n=75,p=0,51,r=0,5,t=650	275,60	970,00	962,90	0,49	52,42	69,93	71,59
3-n=75,p=0,75,r=0,5,t=650	99,10	1137,00	1130,10	0,47	67,31	74,17	91,28
4-n=212,p=0,25,r=0,4,t=1390	1270,60	3380,00	3301,30	1,76	443,75	90,03	62,41
5-n=212,p=0,50,r=0,4,t=1390	1072,90	3139,00	3033,30	1,99	458,06	95,08	65,82
6-n=212,p=0,75,r=0,4,t=1390	-686,00	2529,00	2448,40	1,67	458,00	95,35	127,13
7-n=228,p=0,25,r=0,6,t=1520	1614,50	3460,00	3412,70	0,67	371,76	91,93	53,34
8-n=228,p=0,50,r=0,6,t=1520	1639,20	3248,00	3110,30	1,83	359,86	91,18	49,53
9-n=228,p=0,75,r=0,6,t=1520	-587,90	2577,00	2473,80	2,16	352,69	87,82	122,81
10-n=399,p=0,25,r=0,5,t=2310	2784,80	5668,00	5615,00	0,77	853,29	97,83	50,87
11-n=399,p=0,50,r=0,5,t=2310	2353,70	5054,00	4964,80	1,18	882,27	96,45	53,43
12-n=399,p=0,75,r=0,5,t=2310	-1139,80	3871,00	3778,20	1,63	878,14	96,87	129,44
13-n=499,p=0,25,r=0,5,t=2690	3063,20	6447,00	6326,40	1,08	1182,94	97,92	52,49
14-n=499,p=0,50,r=0,5,t=2690	2395,90	5759,00	5584,20	1,69	1278,46	97,90	58,40
15-n=499,p=0,75,r=0,5,t=2690	-2012,30	4007,00	3762,30	4,35	1255,33	98,00	150,22
16-n=600,p=0,25,r=0,5,t=3096	3930,60	7966,00	7814,10	0,87	1720,42	98,60	50,66
17-n=600,p=0,50,r=0,5,t=3096	2987,90	6916,00	6808,50	1,11	1793,31	97,47	56,80
18-n=600,p=0,75,r=0,5,t=3096	-2140,00	4732,00	4498,70	3,09	1729,97	98,40	145,22
19-n=699,p=0,25,r=0,5,t=3420	4719,90	9371,00	9258,90	0,77	2139,19	98,85	49,63
20-n=699,p=0,50,r=0,5,t=34200	3535,20	8074,00	7845,80	1,93	2311,02	99,03	56,22
21-n=699,p=0,75,r=0,5,t=34201	-1475,10	6299,00	5887,60	3,12	2321,20	98,72	123,42
21-n=800,p=0,25,r=0,6,t=35272	5870,30	11071,00	10940,50	0,93	2650,11	99,30	46,98
22-n=800,p=0,50,r=0,6,t=35273	5283,50	10128,00	9904,60	1,15	2745,97	98,58	47,83
23-n=800,p=0,75,r=0,6,t=35274	-1481,30	7427,00	7215,70	2,84	2680,50	98,92	119,94
24-n=899,p=0,25,r=0,6,t=37465	6992,80	12920,00	12760,50	0,71	3304,29	99,48	45,88
25-n=899,p=0,50,r=0,6,t=37466	6373,20	11599,00	11075,50	2,20	3384,36	99,30	45,05
26-n=899,p=0,75,r=0,6,t=37467	-1161,40	8626,00	8436,40	1,39	3376,32	99,33	113,46
27-n=1000,p=0,25,r=0,6,t=39508	7676,70	14185,00	13978,40	0,64	3809,09	99,43	45,88
28-n=1000,p=0,50,r=0,6,t=39509	7468,80	12671,00	12338,40	1,54	4151,59	99,57	41,06
29-n=1000,p=0,75,r=0,6,t=39500	-1614,00	8393,00	8261,20	1,67	3727,89	99,28	119,23
31-n=1236,p=0,25,r=0,6,t=44311	9458,30	16880,00	16761,30	0,45	5440,50	99,35	43,97
32-n=1236,p=0,50,r=0,6,t=4431	7752,00	14385,00	13988,20	1,49	5749,58	99,35	46,11
33-n=1236,p=0,75,r=0,6,t=4431	-1945,30	10231,00	9996,40	1,78	5470,08	99,32	119,01

Analisando os resultados reportados na Tabela 5.3, observa-se uma tendência de piora da solução inicial, S_0, à medida que a taxa de tarefas prioritárias aumenta para instâncias de mesmo tamanho de tarefa. Após uma análise comparativa entre as estruturas das soluções S_0

e S^*, notou-se que essa piora está ligada a uma limitação da heurística de geração da solução inicial, a qual, apesar de beneficiar as tarefas prioritárias no início da alocação, não garante um bom uso dos períodos de produção não supervisionados nem um bom sequenciamento que minimize as trocas de ferramentas, o que degrada significativamente a solução. Essa deficiência é subsequentemente corrigida pelo PT ao longo de sua execução.

Quanto ao desvio padrão, observa-se uma variação muito baixa entre as execuções do método, com um máximo de $4,35\%$ na instância 14. Esse fato indica a robustez do método na busca pela solução. O *gap* entre a solução inicial e a final também é um indicador de qualidade a se destacar, tendo uma média de 75,79% entre todas as instâncias, evidenciando uma melhora significativa entre a solução inicial e a solução obtida, demonstrando a real efetividade do método. Outra métrica que corrobora a robustez do método é o *gap* entre S e S^*, ou seja, a diferença percentual entre a solução média e a melhor solução, calculada utilizando-se a Equação 5.1, possui uma média de apenas 2,45%. O comportamento do *gap* entre S_0 e S é similar, tendo uma média de 75.47%, muito similar ao *gap* entre S_0 e S^*, reforçando a robustez do PT, novamente.

Outro ponto a ser observado é a taxa de convergência. Com exceção do primeiro grupo de instâncias, essa taxa permaneceu acima de 90%, o que sugere a possibilidade de que o método encontraria soluções melhores caso continuasse em execução, aumentando-se os valores de qT_{emp} e de N_{kmax}. No entanto, é importante notar que o tempo de execução se torna um fator limitante, uma vez que apresenta crescimento rápido, com máximo de 5822 segundos e um mínimo de 39 segundos dentre todas as execuções. Considerando a paralelização em CPU do PT, é lógico assumir que outros métodos em um cenário estritamente sequencial teriam uma performance ainda pior, ao menos em termos de tempo de execução.

6 Conclusão

Neste estudo, foram abordadas a definição formal, uma revisão detalhada da literatura e a descrição de uma metaheurística para a abordagem de uma variante do problema de minimização de trocas de ferramentas, a qual estende o problema base com novas características como período de produção não supervisionado, máquinas paralelas, prioridade entre tarefas, e reentrância de tarefas, entre outras características específicas. Observou-se que este é um problema de relevância significativa no contexto prático industrial, uma vez que a redução das trocas de ferramentas, entre outros objetivos, pode levar a uma diminuição substancial dos custos operacionais e aumentar a eficiência produtiva. Além disso, a revisão da literatura revelou que existe apenas uma única abordagem para a resolução desta versão específica do problema, havendo espaço considerável para contribuições inovadoras, especialmente quando se trata de transparência e reprodutibilidade de experimentos.

Dados os experimentos computacionais, conclui-se que o PT demonstra estabilidade na quantidade de tarefas finalizadas e não finalizadas para cada grupo de instâncias. Apesar da limitação inicial na heurística de geração de soluções, o PT corrige essas deficiências ao longo da execução, proporcionando melhorias significativas, evidenciadas pelo *gap* médio de 75,79%. Além disso, a baixa variação entre execuções reforça a robustez do método. No entanto, a taxa de convergência sugere que soluções ainda melhores poderiam ser encontradas com maior tempo de execução, embora este se torne um fator limitante em cenários mais complexos. Além disso, o cálculo da função de avaliação se mostra um claro gargalo para o método. Dessa maneira pretende-se realizar otimizações e propor abordagens paralelas para a mesma, a fim de melhorar o desempenho do método.

Trabalhos futuros se concentrarão na exploração de um novo conjunto de instâncias mais desafiadoras, caracterizado por operações de uma mesma tarefa que possuem conjuntos de ferramentas diferentes. Esta versão alternativa é relevante, pois aumentará o desafio na resolução do problema, uma vez que impede o agrupamento de operações da mesma tarefa. Além disso, a implementação do modelo matemático será realizada para permitir a obtenção de resultados comparativos. A fim de melhorar a eficiência do método, ou seja, obter uma convergência mais rápida sem desperdício de tempo computacional, um critério de parada adaptativo será desenvolvido, baseado na convergência do método de resolução. Por fim, pretende-se relatar os achados em formato de artigo para o Simpósio Brasileiro de Pesquisa Operacional.

Referências

AGNETIS, A.; DETTI, P.; PRANZO, M.; SODHI, M. Sequencing unreliable jobs on parallel machines. *J. Scheduling*, v. 12, p. 45–54, 02 2009.

ALMEIDA, A. L. B. *Revisitando o Revenimento Paralelo: Computação de Alto Desempenho e Aplicação em Pesquisa Operacional*. Tese (Tese de doutorado) — Universidade Federal de Ouro Preto, Ouro Preto, Agosto 2024.

ALMEIDA, A. L. B.; de Castro Lima, J.; CARVALHO, M. A. M. On serial and parallel evaluation functions for job sequencing and tool switching problems. *Computers & Operations Research*, v. 177, p. 106969, 2025. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054824004416>.

ALMEIDA, A. L. B.; de Castro Lima, J.; CARVALHO, M. A. M. Revisiting the parallel tempering algorithm: High-performance computing and applications in operations research. *Computers & Operations Research*, v. 178, p. 107000, 2025. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054825000280>.

BARD, J. F. A heuristic for minimizing the number of tool switches on a flexible machine. *IIE Transactions*, Taylor & Francis, v. 20, n. 4, p. 382–391, 1988. Disponível em: <https://doi.org/10.1080/07408178808966195>.

BEEZãO, A. C.; CORDEAU, J.-F.; LAPORTE, G.; YANASSE, H. H. Scheduling identical parallel machines with tooling constraints. *European Journal of Operational Research*, v. 257, n. 3, p. 834–844, 2017. ISSN 0377-2217. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0377221716306233>.

CALMELS, D. The job sequencing and tool switching problem: state-of-the-art literature review, classification, and trends. *International Journal of Production Research*, Taylor & Francis, v. 57, n. 15-16, p. 5005–5025, 2019. Disponível em: <https://doi.org/10.1080/00207543.2018.1505057>.

CHCOMPANY the B. R. *Metalworking Machinery Global Market Report 2024 – By Type*. 2024. Accessed: April 15, 2024. Disponível em: <https://www.thebusinessresearchcompany.com/report/metalworking-machinery-global-market-report>.

CRAMA, Y.; KOLEN, A. W. J.; OERLEMANS, A. G.; SPIEKSMA, F. C. R. *Minimizing the Number of Tool Switches on a Flexible Machine*. 1994. 33–54 p. Disponível em: <https://doi.org/10.1007/BF01324874>.

DANG, Q.-V.; HERPS, K.; MARTAGAN, T.; ADAN, I.; HEINRICH, J. Unsupervised parallel machines scheduling with tool switches. *Computers and Operations Research*, v. 160, p. 106361, 2023. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054823002253>.

EARL, D. J.; DEEM, M. W. Parallel tempering: Theory, applications, and new perspectives. *Physical Chemistry Chemical Physics*, Royal Society of Chemistry (RSC), v. 7, n. 23, p. 3910, 2005. ISSN 1463-9084. Disponível em: <http://dx.doi.org/10.1039/B509983H>.

FARUGHI, H.; DOLATABADIAA, M.; MORADI, V.; KARBASI, v.; MOSTAFAYI, S. Minimizing the number of tool switches in flexible manufacturing cells subject to tools reliability using genetic algorithm. *Journal of Industrial and Systems Engineering*, Iranian Institute of Industrial Engineering, v. 10, n. special issue on Quality Control and Reliability, p. 17–33, 2017. ISSN 1735-8272. Disponível em: <https://www.jise.ir/article_33655.html>.

FOGLIATTO, F. S.; da Silveira, G. J.; BORENSTEIN, D. The mass customization decade: An updated review of the literature. *International Journal of Production Economics*, v. 138, n. 1, p. 14–25, 2012. ISSN 0925-5273. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925527312000989>.

GAN, Z. L.; MUSA, S. N.; YAP, H. J. A review of the high-mix, low-volume manufacturing industry. *Applied Sciences*, v. 13, n. 3, 2023. ISSN 2076-3417. Disponível em: <https://www.mdpi.com/2076-3417/13/3/1687>.

GHRAYEB, N. P. O. A.; FINCH, P. R. A mathematical model and heuristic procedure to schedule printed circuit packs on sequencers. *International Journal of Production Research*, Taylor & Francis, v. 41, n. 16, p. 3849–3860, 2003. Disponível em: <https://doi.org/10.1080/0020754031000118071>.

HANSMANN, U. H. Parallel tempering algorithm for conformational studies of biological molecules. *Chemical Physics Letters*, Elsevier BV, v. 281, n. 1–3, p. 140–150, dez. 1997. ISSN 0009-2614. Disponível em: <http://dx.doi.org/10.1016/S0009-2614(97)01198-6>.

HIRVIKORPI, M.; SALONEN, K.; KNUUTILA, T.; NEVALAINEN, O. S. The general two-level storage management problem: A reconsideration of the ktns-rule. *European Journal of Operational Research*, v. 171, n. 1, p. 189–207, 2006. ISSN 0377-2217. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0377221704005934>.

INDUSTRYARC. *Printed Circuit Board Assembly Market - Forecast(2024 - 2030)*. 2024. Accessed: April 15, 2024. Disponível em: <https://www.industryarc.com/Report/18291/printed-circuit-boards-pcb-assembly-market.html>.

KEUNG, K. W.; IP, W. H.; LEE, T. C. The solution of a multi-objective tool selection model using the ga approach. *The International Journal of Advanced Manufacturing Technology*, v. 18, n. 11, p. 771–777, 2001. ISSN 1433-3015. Disponível em: <https://doi.org/10.1007/s001700170001>.

LÓPEZ-IBÁÑEZ, M.; DUBOIS-LACOSTE, J.; CÁCERES, L. P.; BIRATTARI, M.; STÜTZLE, T. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, Elsevier, v. 3, p. 43–58, 2016.

MATZLIACH, B.; TZUR, M. Storage management of items in two levels of availability. *European Journal of Operational Research*, v. 121, n. 2, p. 363–379, 2000. ISSN 0377-2217. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0377221799000375>.

MECLER, J.; SUBRAMANIAN, A.; VIDAL, T. A simple and effective hybrid genetic search for the job sequencing and tool switching problem. *Computers and Operations Research*, v. 127, p. 105153, 11 2020.

METROPOLIS, N.; ROSENBLUTH, A. W.; ROSENBLUTH, M. N.; TELLER, A. H.; TELLER, E. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, v. 21, n. 6, p. 1087–1092, 06 1953. ISSN 0021-9606. Disponível em: <https://doi.org/10.1063/1.1699114>.

MORGAN, J.; HALTON, M.; QIAO, Y.; BRESLIN, J. G. Industry 4.0 smart reconfigurable manufacturing machines. *Journal of Manufacturing Systems*, v. 59, p. 481–506, 2021. ISSN 0278-6125. Disponível em: <https://www.sciencedirect.com/science/article/pii/S027861252100056X>.

NOëL, M.; SODHI, M. S.; LAMOND, B. F. Tool planning for a lights-out machining system. *Journal of Manufacturing Systems*, v. 26, n. 3, p. 161–166, 2007. ISSN 0278-6125. Design, Planning, and Control for Reconfigurable Manufacturing Systems. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0278612508000344>.

PRIVAULT, C.; FINKE, G. Modelling a tool switching problem on a single nc-machine. *Journal of Intelligent Manufacturing*, v. 6, n. 2, p. 87–94, abr. 1995. ISSN 1572-8145. Disponível em: <https://doi.org/10.1007/BF00123680>.

QIU, Y.; CHERNIAVSKII, M.; GOLDENGORIN, B.; PARDALOS, P. M. A computational study of the tool replacement problem. *INFORMS Journal on Computing*, INFORMS, 2025.

RUPE, J.; KUO, W. Solutions to a modified tool loading problem for a single fmm. *International Journal of Production Research*, Taylor & Francis, v. 35, n. 8, p. 2253–2268, 1997. Disponível em: <https://doi.org/10.1080/002075497194831>.

SPLUNK. *Cyber-Physical Systems (CPS) Explained*. 2024. Accessed: April 22, 2024. Disponível em: <https://www.splunk.com/en_us/blog/learn/cyber-physical-systems.html>.

TANG, C. S.; DENARDO, E. V. Models arising from a flexible manufacturing machine, part i: Minimization of the number of tool switches. *Operations Research*, v. 36, n. 5, p. 767–777, 1988. Disponível em: <https://doi.org/10.1287/opre.36.5.767>.

TZUR, M.; ALTMAN, A. Minimization of tool switches for a flexible manufacturing machine with slot assignment of different tool sizes. *IIE Transactions*, v. 36, p. 95–110, 02 2004.

ÖZPEYNIRCI, S.; GöKGüR, B.; HNICH, B. Parallel machine scheduling with tool loading. *Applied Mathematical Modelling*, v. 40, n. 9, p. 5660–5671, 2016. ISSN 0307-904X. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0307904X16000093>.