57º SBPO
SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL
Gramado • RS
DE 5 A 9 DE OUTUBRO DE 2025

ALGORITMO GENÉTICO PARA MINIMIZAÇÃO DE DESPERDÍCIO DE MATÉRIA-PRIMA EM UM PROBLEMA DE FLOW SHOP SCHEDULING COM RESTRIÇÕES DE NO-WAIT E SETUPS ANTECIPADOS

Marta Ferreira de Freitas
Universidade Federal de Campina Grade - UFCG
Rua Luiz Grande, S/N, Frei Damião, Sumé, Paraíba - 58540-000
martamff12@gmail.com

Paloma dos Santos Alves Nunes
Universidade Federal da Paraíba - UFPB
Conj. Pres. Castelo Branco III, João Pessoa - PB, 58033-455
pnunes.pn123@gmail.com

Yuri Laio Teixeira Veras Silva
Universidade Federal de Campina Grade - UFCG
Rua Luiz Grande, S/N, Frei Damião, Sumé, Paraíba - 58540-000
yurilaio@gmail.com

RESUMO

O problema de *Flow Shop Scheduling* (FSS) é amplamente estudado devido à sua relevância em ambientes produtivos, como a indústria alimentícia. Este trabalho aborda uma variação emergente do FSS, visando minimizar o desperdício de matéria-prima sob restrições de no-wait, setups antecipados e capacidade limitada das máquinas. Foi proposto um Algoritmo Genético (GA) adaptado para resolver o problema, incorporando mecanismos específicos ao contexto industrial. O modelo foi validado em oito instâncias reais de uma grande produtora de laticínios brasileira, comparando-se seus resultados com os da solução empregada pela empresa. Os experimentos demonstraram que o GA supera a abordagem atual em qualidade de solução, mantendo tempos computacionais viáveis para operações reais, evidenciando o potencial da abordagem proposta na redução perdas em processos produtivos com as restrições consideradas.

PALAVRAS CHAVE. Flow Shop Scheduling. Algoritmo Genético. Indústria alimentícia.

Tópicos (PO na Administração e Gestão da Produção, PO na Indústria)

ABSTRACT

The Flow Shop Scheduling (FSS) problem is widely studied due to its industrial significance, particularly in food production systems. This work investigates an emerging FSS variant that minimizes raw material waste under no-wait constraints, anticipatory setups, and limited machine capacity. We propose a customized Genetic Algorithm (GA) incorporating industry-tailored optimization mechanisms. The model was validated using eight real-world instances from a major Brazilian dairy producer, with performance benchmarked against the company's current scheduling solution. Experimental results demonstrate that the GA outperforms the existing approach in solution quality while maintaining computationally feasible processing times. These findings highlight the approach potential for waste reduction in constrained production environments with the specified restrictions.

KEYWORDS. Flow Shop Scheduling. Genetic Algorithm. Food Industry.

Paper topics (OR in Administration and Production Management, OR in Industry)

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br

1. Introdução

No cenário econômico atual, a busca pela otimização dos processos produtivos tem se tornado essencial para as empresas que desejam aumentar sua eficiência e reduzir custos operacionais. A gestão eficiente dos recursos produtivos, especialmente em ambientes industriais complexos, que envolvem múltiplas máquinas e etapas de processamento, é um desafio constante. Dentre os diversos fatores que impactam essa gestão, a definição da ordem de processamento e do tempo de execução das atividades desempenham um papel fundamental na organização do fluxo de trabalho e no uso eficiente dos recursos disponíveis [Zhao et al. 2023].

O sequenciamento da produção (*scheduling*) é um elemento essencial na gestão de operações industriais, pois determina a ordem e o tempo de execução das atividades produtivas para otimizar a eficiência operacional [Silva et al., 2018]. Fatores como produtividade, custos, desperdícios e tempo de resposta da produção, são diretamente influenciados pela tomada de decisão sobre a sequência de processamento. Dessa forma, o estudo e aprimoramento de métodos de *scheduling* têm recebido cada vez mais atenção tanto na academia quanto na indústria, uma vez que sistemas produtivos eficientes são cruciais para a competitividade empresarial [Aqil 2024].

No setor alimentício, o sequenciamento da produção assume um papel ainda mais crítico devido à natureza perecível dos insumos, à necessidade de cumprir normas sanitárias rigorosas e o alto teor de variabilidade na demanda. A produção muitas vezes ocorre em sistemas do tipo *flow shop*, nos quais um conjunto de tarefas deve ser processado em uma sequência fixa de máquinas ou etapas produtivas [Tangour et al. 2021]. Nesse contexto, Conserva e Silva [2024] destacam que um planejamento inadequado pode levar à ociosidade de equipamentos, desperdício de matéria-prima e aumento do tempo de espera entre operações, comprometendo a produtividade e elevando os custos operacionais.

O desperdício de matéria-prima na indústria alimentícia representa um problema significativo. De acordo com o Programa das Nações Unidas para o Meio Ambiente [PNUMA 2024], globalmente, cerca de 13,2% dos alimentos são perdidos na cadeia de suprimentos entre a colheita e os estágios de varejo. No ambiente fabril, essas perdas podem estar associadas a uma programação ineficiente da produção, que pode levar a deterioração dos insumos, especialmente em processos que envolvem produtos frescos ou sensíveis a variações de temperatura. Além de elevar os custos operacionais, essas ineficiências comprometem a capacidade das empresas de atender à demanda de forma competitiva. Diante desse cenário, a busca por técnicas de otimização capazes de minimizar desperdícios e melhorar a eficiência produtiva torna-se essencial.

Em geral, problemas de *flow shop scheduling* são resolvidos por métodos exatos e heurísticos, mas essas abordagens podem ser ineficazes diante da complexidade crescente dos sistemas produtivos modernos, especialmente em cenários de grande escala. Nesse sentido, o uso de metaheurísticas tem se destacado como uma alternativa promissora para encontrar soluções eficientes em um tempo computacional viável [Goli et al. 2023; Souza 2023].

Nesse contexto, este estudo tem como objetivo desenvolver e implementar um algoritmo genético para otimizar o problema de *flow shop scheduling* heterogêneo com múltiplos estágios no setor alimentício. Os testes serão realizados em instâncias de dados fornecidas por uma empresa de laticínios que produz diferentes tipos de queijos. O foco principal é minimizar o consumo de matéria-prima, garantindo o atendimento integral da demanda dentro dos prazos estabelecidos.

2. Revisão da Literatura

Um dos primeiros autores a abordar formalmente o problema de *Flow Shop Scheduling* (FSS) na literatura acadêmica foi Johnson [1954], onde o autor propôs a conhecida “Regra de Johnson” para o sequenciamento de tarefas em um sistema de produção com duas máquinas, visando minimizar o *makespan*. Com o avanço das pesquisas, diferentes variações do problema de FSS passaram a ser estudadas. No caso do problema de *Flow Shop* com máquinas paralelas e tempos de setup dependentes da sequência, Garavito-Hernández et al. [2019] aplicaram o *Imperialist Competitive Algorithm* (ICA) e o Algoritmo Genético (GA). O experimento foi conduzido em doze instâncias e os resultados foram comparados com um modelo de programação

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br

inteira mista, onde a meta-heurística obteve soluções iguais ou melhores que o método exato em metade das instâncias.

Ainda se tratando de métodos heurísticos, Chung e Liao [2013] propuseram um algoritmo baseado em Sistema Imunológico Artificial (AIS) para um *Flow Shop* Híbrido (HFS), comparando-o com algoritmos baseados e não baseados em sistemas imunes. Para o mesmo tipo de problema Yu, Semeraro e Matta [2018] propuseram um Algoritmo Genético (GA) com máquinas paralelas não relacionadas e restrições de elegibilidade, minimizando o tempo total de atraso, sendo comparado a diversos algoritmos, em 450 instâncias com diferentes tamanhos e padrões de correlação dos tempos de processamento das operações.

Da mesma forma, Zhao et al. [2022] desenvolveram uma heurística construtiva baseada em conhecimento combinada com uma meta-heurística de três estágios para resolver o FSS com blocagem e tempos de setup dependente da sequência (SDST/DBFSP), validado em 2880 instâncias de *benchmark* e 10 conjuntos reais da indústria de alumínio.

Recentemente, Anunay et al. [2023] desenvolveram modelos exatos MILP para o problema *Multi-Stage Hybrid Assembly Flow Shop* com Máquinas Paralelas Idênticas, buscando otimizar o *makespan* e o agendamento da produção com instâncias reais. No mesmo ano, Sadki et al. [2023] propuseram um modelo MILP para o problema de *Permutation Flow Shop Scheduling* (PFSSP), considerando tempos de *setup* independentes da sequência. A validação do modelo foi realizada por meio de comparações com os métodos GA, ILS e IG.

3. Descrição do problema

Nos sistemas produtivos do tipo *Flow Shop*, as tarefas seguem uma sequência predefinida de processamento em um conjunto de máquinas dispostas em série. Esse tipo de arranjo é comumente encontrado em setores industriais que necessitam de alta eficiência e previsibilidade no fluxo de produção.

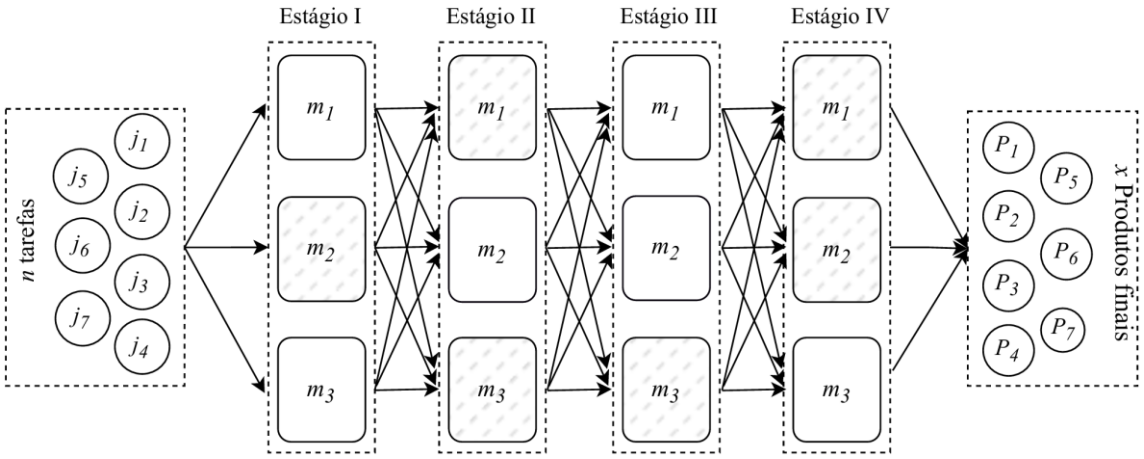
Diante disso, o presente estudo investiga o problema de *Flow Shop Scheduling with Raw Waste Consumption Minimization* (FSS-RWC), em um ambiente de *flow shop* com múltiplas máquinas heterogêneas, aplicado em uma empresa de laticínios que produz diferentes tipos de queijos (coalho, minas e prato). O objetivo central é minimizar o consumo de matéria-prima, assegurando que toda a demanda seja atendida dentro dos prazos estabelecidos, enquanto respeitam as restrições operacionais fundamentais.

Esse objetivo foi definido em razão da sua grande importância no setor alimentício, onde o desperdício de matéria-prima impacta diretamente os custos e a eficiência da produção. Minimizar esse consumo garante um melhor aproveitamento dos insumos, evita perdas e assegura que a demanda seja atendida de forma precisa, contribuindo para a competitividade do negócio. Isto é alcançado através da definição de uma sequência ótima de produção que reduz perdas nos tempos de *setup* e evita desperdícios decorrentes de trocas frequentes entre produtos distintos. Para tal, torna-se fundamental considerar um balanceamento eficiente da ordem de produção, garantindo o atendimento das restrições operacionais do sistema.

No problema em questão, são considerados quatro estágios de produção (como ilustrado na Figura 1), onde os diferentes tipos de queijo serão processados. Cada estágio possui três máquinas heterogêneas que apresentam características distintas, como capacidade de processamento, eficiência, consumo de recursos e tempos de configuração (setup). A organização realiza planejamentos de produção quinzenais, onde a quantidade de pedidos totais a serem processados varia de acordo com a demanda de cada período.

Figura 1 – Ilustração de sistema FSS com múltiplas máquinas heterogêneas

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br



Fonte: Elaborado pelo autor (2025).

Em ambientes de produção de produtos altamente perecíveis, como o de laticínios, os insumos precisam de condições específicas de sequenciamento que preservem a qualidade final do produto. Uma das especificações a serem seguidas é que, uma vez iniciado o processamento de um pedido, ele deve ser transferido de um estágio a outro sem qualquer tempo de espera intermediário (*no-wait*). Interrupções entre os estágios podem comprometer irreversivelmente o produto final, provocando alterações no sabor, na textura e na qualidade do laticínio.

Em decorrência direta dessa restrição, torna-se necessário que o tempo de preparação das máquinas (*setup*) seja feito sempre de forma antecipada, evitando esperas entre os estágios. Sabendo que cada máquina pode processar apenas uma tarefa por vez e que no sistema em questão serão processados três diferentes tipos de queijo, o *setup* precisa ser realizado sempre que houver a troca entre esses diferentes tipos de pedidos na mesma máquina. É importante ressaltar que este tempo não é fixo, podendo depender da sequência em que as tarefas são processadas (o tempo necessário para preparar uma máquina *m* antes de processar uma nova tarefa *j* depende da tarefa anterior na sequência), das características técnicas da máquina e da compatibilidade entre os insumos utilizados.

4. Algoritmo proposto

O Algoritmo Genético (GA) é uma metaheurística de busca inspirada no processo de evolução biológica, fundamentada no princípio da "sobrevivência do mais apto" [Ilin et al. 2023]. Desse modo, o objetivo do GA consiste em identificar o indivíduo com a melhor qualidade dentro do espaço de busca. A qualidade de cada indivíduo é avaliada por meio de uma função de aptidão.

A primeira etapa de desenvolvimento do GA envolve a criação de uma população inicial de soluções, que são aprimoradas por meio de um processo evolutivo. Os indivíduos com maior aptidão são selecionados como pais, com base em uma função de aptidão maximizada e geram descendentes pelo operador de *crossover*, que permite a herança genética [Guan et al. 2023]. O operador de mutação introduz mudanças aleatórias nos descendentes, possibilitando a exploração de novas soluções. O processo se repete até que uma solução satisfatória seja encontrada. Além disso, os parâmetros podem ser ajustados para otimizar o algoritmo conforme o problema.

Dentre os parâmetros estabelecidos para a inicialização do GA desenvolvido, destacam-se o tamanho da população P, o número máximo de gerações max_iter, a taxa de mutação mínima (T_{min}) e a taxa de mutação máxima (T_{max}). O parâmetro P define o número de indivíduos que compõem a população em cada geração, impactando diretamente a diversidade genética e a velocidade de convergência do algoritmo. Já o número máximo de iterações atua como um critério de parada no algoritmo, definindo quantas populações devem ser geradas no total.

A mutação, por sua vez, introduz variações aleatórias nos cromossomos, alterando um ou mais de seus genes, sendo essencial para manter a diversidade genética da população. Nesse trabalho, foi adotada uma estratégia de mutação com taxa adaptativa, onde T_{min} e T_{max} funcionam

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br

para controlar o balanceamento entre a diversificação e a convergência do algoritmo. A Tabela 1 apresenta os valores empregues neste trabalho para os parâmetros do GA.

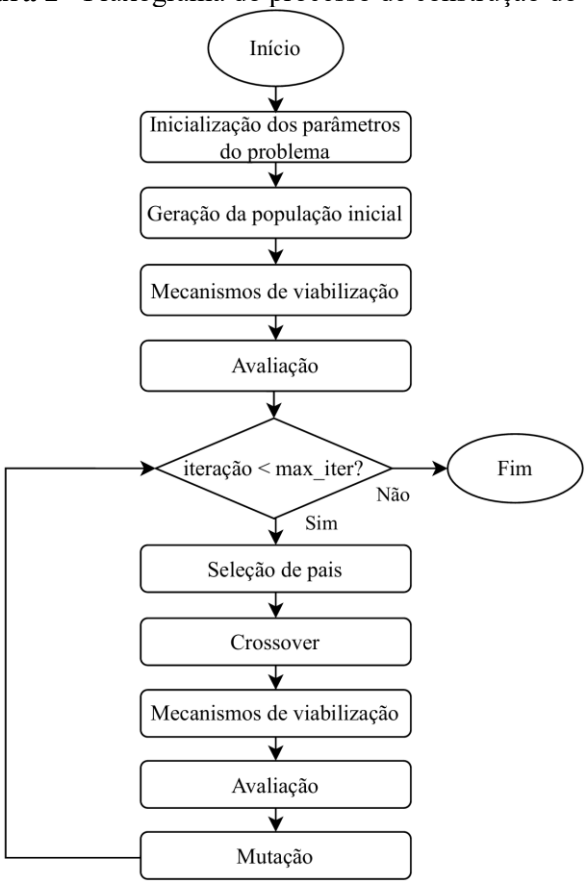
Tabela 1 - Valores dos parâmetros adotados no método proposto

Parâmetro	Símbolo	Valor
Tamanho da população de indivíduos	*P*	200
Número de iterações	*max_iter*	1000
Taxa de mutação máxima	*Tmax*	0.25
Taxa de mutação mínima	*Tmin*	0.08

Fonte: Elaborado pelo autor (2025).

No decorrer da execução do algoritmo, uma população de soluções orientada pelo critério de aptidão, evolui ao longo das gerações através dos operadores de seleção, cruzamento e mutação. A partir disso, forma-se uma nova geração de indivíduos que substitui a população anterior. A Figura 2 ilustra o fluxograma de construção do algoritmo genético proposto.

Figura 2 - Fluxograma do processo de construção do GA



Fonte: Elaborado pelo autor (2025).

4.1 Representação da solução

Tendo em vista que o algoritmo genético opera com uma população de indivíduos que representam soluções potenciais para o problema em questão, torna-se fundamental criar uma estrutura de representação que se alinhe às características do ambiente *flow shop* analisado.

No contexto do problema de FSS-RWC, que envolve quatro estágios de produção, cada um com três máquinas paralelas, e um conjunto de *n* pedidos de produção, optou-se por uma

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br

representação em forma de matriz tridimensional. Essa estrutura não apenas reflete a sequência de produção, mas também a alocação de cada pedido a uma máquina específica em cada estágio.

Conforme ilustrado nas Tabelas 2 e 3, a solução é composta pelo sequenciamento de pedidos (um vetor de tamanho n, que representa a ordem em que os pedidos serão processados no primeiro estágio) e a alocação de máquinas (uma matriz de dimensão $k \times j$, onde cada linha corresponde a um estágio e cada coluna a um pedido, indicando qual máquina será utilizada para processar o pedido j no estágio k).

Tabela 2 – Representação do vetor sequência

j_3	j_5	j_0	j_1	j_6	j_7	j_4	j_2

Fonte: Elaborado pelo autor (2025).

Tabela 3 – Representação da matriz de alocação

k_0	m_1	m_1	m_3	m_2	m_1	m_3	m_2	m_1
k_1	m_1	m_2	m_1	m_2	m_2	m_3	m_1	m_1
k_2	m_1	m_1	m_3	m_1	m_2	m_3	m_2	m_1
k_3	m_3	m_2	m_3	m_1	m_1	m_1	m_2	m_3

Fonte: Elaborado pelo autor (2025).

4.2 Inicialização

A inicialização do GA se dá com a criação aleatória da primeira população de indivíduos de tamanho P. Cada indivíduo simboliza uma solução para o problema em questão, ao longo dos quatro estágios de produção.

Como citado anteriormente, a solução é representada por um vetor unidimensional que indica uma permutação aleatória dos pedidos e por uma matriz tridimensional onde cada componente representa uma máquina atribuída para cada pedido em cada estágio. Na fase de inicialização, o algoritmo confere se todos os pedidos foram alocados exatamente uma vez em cada etapa, sem exceder a capacidade de processamento das máquinas. Para isso, o vetor permutação é inicialmente preenchido com todos os pedidos e embaralhado aleatoriamente, o que garante que não existam repetições. Assim gera-se uma matriz que expressa as decisões de alocação de cada pedido em cada etapa e máquina, conforme demonstrado no Algoritmo 1.

Algoritmo 1 - Inicialização

```
1: Procedimento: Geração da solução inicial
2: para cada indivíduo i na população faça
3:     para cada elemento j do indivíduo faça
4:         solução[i][j] = j
5:     Embaralhe aleatoriamente os pedidos da sequência do indivíduo i
6:     para cada estágio k faça
7:         para cada elemento j do indivíduo faça
8:             Aloque aleatoriamente o pedido j a uma das máquinas m disponíveis no estágio k
```

Após definir as sequências e alocações, o processo de viabilização é executado para ajustar os tempos de processamento e assegurar que as restrições sejam cumpridas.

4.3 Mecanismos de viabilização

Durante o processo evolutivo do GA, os operadores genéticos podem gerar cromossomos que, na prática, não são viáveis para a resolução do problema proposto. Isso ocorre devido ao fato

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br

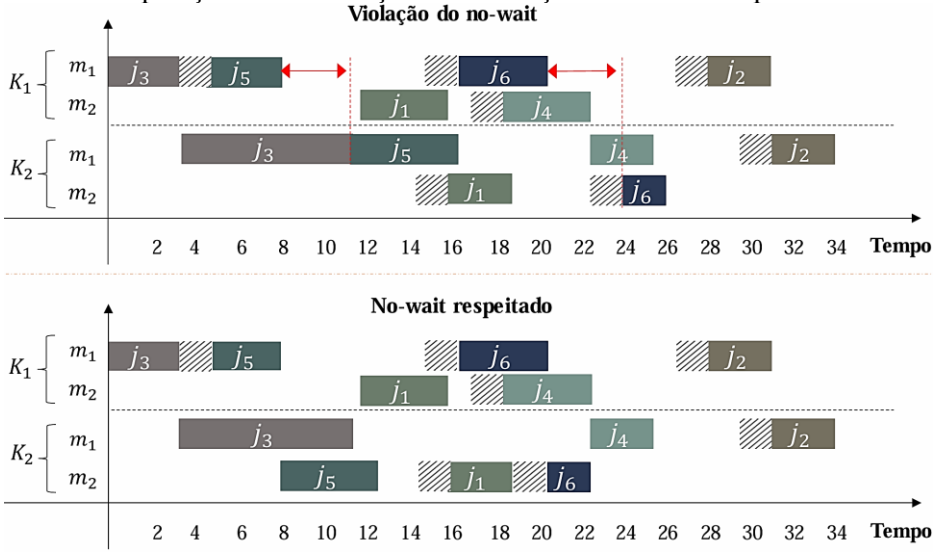
de serem consideradas diferentes restrições operacionais. Na indústria alimentícia, em especial, erros de sequenciamento podem comprometer seriamente a qualidade dos produtos, gerar desperdícios de matéria-prima e causar atrasos no processo produtivo.

Diante disso, torna-se necessária a implementação de mecanismos de viabilização para garantir que apenas indivíduos factíveis sejam mantidos na população. Esses mecanismos realizam os ajustes necessários na estrutura dos indivíduos para que os critérios de viabilidade definidos sejam integralmente atendidos.

No que se refere a restrição de no-wait, o algoritmo assegura que o processamento de uma tarefa em um determinado estágio inicie imediatamente após a sua conclusão no estágio anterior. Durante a construção do código, são verificados os tempos de início e conclusão das tarefas em cada estágio para que assim seja possível identificar se a restrição está sendo atendida. Quando necessário, o algoritmo introduz atrasos intencionais nos estágios anteriores, ajustando os tempos de início e conclusão, para que o fluxo siga sem interrupções. De forma integrada, também é tratado o tempo de setup: antes de processar um pedido em uma máquina, verifica-se se o tipo é diferente do último executado; em caso positivo, um tempo adicional de preparação (tempo de *setup*) é incorporado, garantindo que a troca não comprometa a restrição de no-wait. Esse cálculo considera simultaneamente a conclusão do estágio anterior e a disponibilidade da máquina, assegurando a sincronização entre preparação e processamento. Assim, a abordagem evita paradas não planejadas e mantém a viabilidade da solução.

Para melhor compreensão, a Figura 3 ilustra dois gráficos de Gantt, onde o primeiro representa um cenário em que a restrição é violada, resultando em uma espera entre dois estágios, e o segundo representa o problema corrigido.

Figura 3 – Comparação entre a solução com violação de no-wait e após sua viabilização

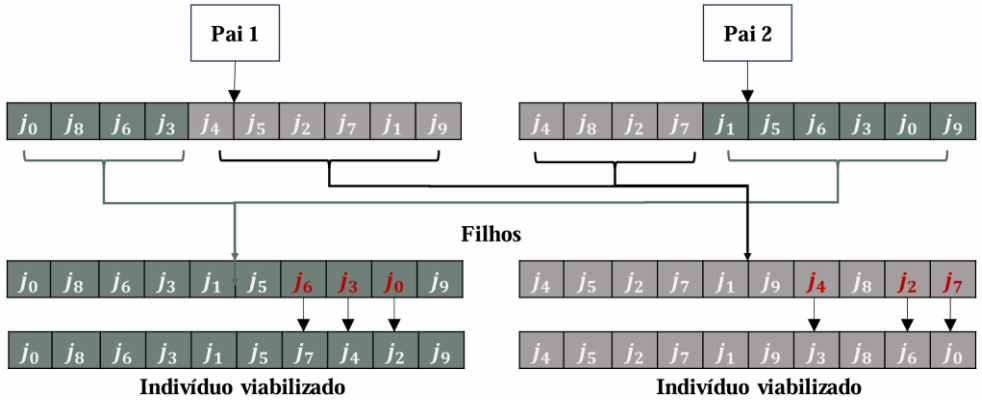


Fonte: Elaborado pelo autor (2025).

Outro ponto importante é que, durante o *crossover* da sequência de pedidos, existe a possibilidade de que a combinação dos pais resulte em cromossomos que apresentam pedidos de produção repetidos. Para lidar com isso, o algoritmo inicializa o vetor correspondente ao filho com valores -1 e realiza uma verificação: cada elemento da segunda parte do cromossomo só é inserido se ainda não estiver presente. Após essa etapa, as posições restantes invalidadas (marcadas com valor -1) são preenchidas com os pedidos que ainda não apareceram, garantindo uma permutação (esse processo está descrito no algoritmo 2, a seguir). Na Figura 4, é possível visualizar o cenário onde essa duplicidade ocorre e sua correção.

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br

Figura 4 - Correção de duplicidade de pedidos pós crossover



Fonte: Elaborado pelo autor (2025)

Os mecanismos para o *no-wait* são aplicados após a geração inicial e após a aplicação dos operadores genéticos. Já o mecanismo para viabilizar na etapa do *crossover* é chamado apenas no *loop* iterativo, garantindo soluções que respeitam as necessidades operacionais do sistema.

4.4 Seleção e cruzamento dos indivíduos

A etapa de seleção visa identificar os indivíduos da população que irão se reproduzir (cromossomos-pais). Inicialmente, a cada nova iteração do GA, é realizado o processo de seleção por torneio, onde dois indivíduos são escolhidos aleatoriamente, e o que apresenta maior aptidão é selecionado como pai para o cruzamento. Isso se repete até que o segundo pai seja selecionado, certificando-se que seja distinto do primeiro, para garantir a diversidade genética.

O operador de *crossover* utilizado é conhecido na literatura como *crossover* de um ponto. A lógica aplicada para o problema consiste, inicialmente, em escolher um ponto de ruptura aleatório no primeiro pai selecionado e copiar esses elementos. Em seguida, as demais posições do vetor são preenchidas com elementos localizados depois do ponto de corte do segundo pai, de modo que a ordem original seja respeitada. Assim, esse vetor passa por um processo de viabilização garantindo que não existam duplicações nas tarefas ordenadas.

Como representado no Algoritmo 2, também é realizado o *crossover* de alocação das máquinas, onde as características entre os genes herdados e a estrutura produtiva são respeitados. Para cada estágio, o filho herda a alocação correspondente aos pontos de corte de cada pai.

Algoritmo 2 – Procedimento de Crossover

```
1: Inicialize vetor filho com tamanho n e valores -1
2: Copie os elementos da sequência do Pai 1 até o ponto de corte para o filho
3: Para os elementos restantes da sequência faça
4:    Verifique se o elemento do Pai 2 já está presente no filho
5:    Se não estiver presente então
        Insira o elemento no filho
6: Para cada elemento i do vetor filho faça
7:       Verifique as tarefas que não foram locadas
8: Para cada elemento i vetor filho faça
9:       Substitua pelo próximo valor da lista de faltantes
10: Atribua o vetor filho como a nova sequência do descendente gerado
11: Para cada pedido j faça
      Para cada estágio k faça
12:       Se j < ponto de corte então
13:          Copie a alocação do Pai 1
14:       Senão
15:          Copie a alocação do Pai 2
```

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br

4.5 Avaliação da solução

A avaliação da solução ocorre através do cálculo da aptidão, realizado com base na eficiência acumulada ponderada de cada pedido. Inicialmente, são consideradas a eficiência de cada pedido nas máquinas as quais foram alocados. É definido para cada uma delas, um valor base de eficiência igual a 1.0. Posteriormente, em todas as etapas de processamento, multiplica-se esse valor pelas eficiências reais atribuídas a cada tarefa, em cada máquina, em cada estágio. Esses valores reais são descritos na instância de dados coletada. A multiplicação acumulada descrita anteriormente resulta em um valor de eficiência global para cada tarefa.

Em seguida, realiza-se uma média ponderada a partir da eficiência acumulada de cada pedido. Nessa etapa, cada eficiência é multiplicada pelo peso de cada pedido. A avaliação prioriza soluções que maximizam o aproveitamento dos recursos produtivos, dado que o objetivo do problema é minimizar o desperdício de matéria-prima nos processos.

Algoritmo 3 – Avaliação da Solução

```
1: para cada estágio k faça
2:    para cada elemento j do individuo faça
3:       Verifique a máquina na qual o pedido j foi alocado
4:       Verifique a eficiência do pedido j no estágio k na máquina m
5: para cada elemento j do indivíduo faça
6:       Calcule a soma ponderada dos pesos e a soma dos pesos
```

4.6 Mutação

A mutação no GA é responsável por evitar que o processo evolutivo apresente convergência prematuramente para ótimos locais, inserindo diversidade genética a população de indivíduos. Em um problema complexo, como o abordado em questão, esse procedimento torna-se indispensável para garantir que novas regiões do espaço de soluções sejam exploradas.

Neste trabalho, foi definida uma probabilidade de mutação adaptativa que decresce linearmente ao longo das iterações, promovendo maior diversidade nos estágios iniciais da execução (favorecendo uma maior diversidade) e uma menor diversidade nos estágios finais. Para isso, foi definida uma taxa de probabilidade de mutação mínima (T_{min}) e uma máxima (T_{max}), com o objetivo de controlar o balanceamento entre a diversificação e a convergência do algoritmo.

Inicialmente, realiza-se um sorteio para cada indivíduo da população de um valor entre 0 e 1. Nos casos em que esse valor é menor ou igual à probabilidade de mutação, o indivíduo em questão é selecionado para sofrer pequenas perturbações. Nessa situação, também foi previamente definido que o total de mutações a serem realizadas são 10% do total de pedidos sequenciados.

No sequenciamento dos pedidos de produção, os indivíduos selecionados têm suas posições trocadas aleatoriamente, reordenando a forma como os pedidos serão processados. Na matriz de alocações, são escolhidos aleatoriamente um pedido e um estágio produtivo, que terão sua respectiva máquina designada substituída por outra escolhida aleatoriamente dentre as disponíveis. Essas mudanças do procedimento de mutação buscam encontrar combinações que possam reincidir em maior eficiência operacional, contribuindo para que a exploração do espaço de busca seja feita de forma eficiente. O Algoritmo 4 melhor descreve o mecanismo adotado.

Algoritmo 4 – Procedimento de Mutação

```
1: para cada indivíduo i da população faça
2:    gere um número aleatório entre 0 e 1
3:    se o número for menor ou igual à taxa de mutação então
4:       calcule o número de mutações como 10% do total de pedidos
5:       para cada mutação m_u faça
6:          escolha e troque duas posições aleatórias na sequência
8:          escolha aleatoriamente um pedido j e um estágio k
9:          realize nova alocação do pedido j para uma máquina aleatória m no estágio k
```

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br

5. RESULTADOS EXPERIMENTAIS

Esta seção apresenta os resultados obtidos com a implementação da abordagem heurística em C++ nas instâncias pertencentes a empresa de produção de laticínios. Os testes experimentais foram executados em um computador equipado com um processador Intel Core i5 de 2,5 GHz, 4 GB de memória RAM e com um sistema operacional Windows 10.

O GA desenvolvido foi aplicado a oito instâncias de dados reais fornecidas pela empresa de laticínios em estudo, cujo planejamento de produção é realizado quinzenalmente. Cada instância corresponde a um período de 15 dias consecutivos, onde foram considerados os dados relativos aos meses de janeiro, fevereiro, março e abril de 2025.

Como discutido anteriormente, a empresa produz três tipos de queijo (coalho, minas e prato), aqui classificados como Tipo 1, Tipo 2 e Tipo 3 respectivamente. Além da distribuição percentual entre os tipos de pedidos, na Tabela 4, estão representadas a quantidade de máquinas disponíveis por estágio e o número de estágios presente em cada instância analisada, refletindo a estrutura física e a capacidade produtiva da empresa no período correspondente.

Tabela 4 – Dados das instâncias

Instância	Período	Nº de Pedidos	Tipo 1	Tipo 2	Tipo 3	Nº de Estágios	Máquinas/Estágio
1	01/01 a 15/01	97	42%	29%	29%	4	3
2	16/01 a 31/01	115	37%	33%	30%	4	3
3	01/02 a 15/02	90	47%	30%	23%	4	3
4	16/02 a 01/03	102	43%	31%	25%	4	3
5	02/03 a 16/03	92	40%	26%	34%	4	3
6	17/03 a 31/03	109	38%	34%	28%	4	3
7	01/04 a 15/04	95	29%	37%	34%	4	3
8	16/04 a 30/04	120	48%	24%	28%	4	3

Fonte: Elaborado pelos autores (2025).

Para cada uma das instâncias foram realizadas 10 execuções, totalizando 80 experimentos. Em cada execução, foi estabelecido um limite de 1000 iterações no *loop* iterativo do algoritmo.

5.1 Resultados computacionais para as instâncias

Na Tabela 5 são apresentados os resultados computacionais obtidos para cada instância. Os valores exibidos incluem o melhor valor de aptidão (menor consumo de matéria-prima), a média dos resultados encontrados nas execuções para cada instância, o valor de eficiência correspondente ao método atualmente utilizado pela organização, o *gap* percentual entre o melhor valor obtido pelo GA e o valor da organização, bem como o tempo médio de execução em segundos.

Tabela 5 – Resultados computacionais obtidos

Instância	GA (Melhor)	GA (Média)	Organização	Gap (%)	Tempo de Execução (s)
1	0,6161	0,6101	0,5336	15,46%	51,67
2	0,6130	0,6056	0,5205	17,77%	64,83
3	0,6307	0,6237	0,5472	15,26%	50,04
4	0,6222	0,6132	0,5381	15,63%	57,50
5	0,6216	0,6154	0,5362	15,93%	51,34
6	0,6153	0,6109	0,5277	16,60%	59,43
7	0,6200	0,6161	0,5294	17,11%	50,49
8	0,6087	0,6015	0,5121	18,86%	75,05

Fonte: Elaborado pelos autores (2025).

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br

Conforme demonstra a Tabela 5, os valores obtidos pelo Algoritmo Genético superaram os resultados do método atualmente utilizado pela organização em todas as instâncias avaliadas. O melhor desempenho encontrado foi o da instância três, com o maior valor fitness (0,6307), enquanto a instância 8 apresentou o menor valor (0,6087). Isso pode ser justificado pelo fato dessa instância concentrar o maior número de pedidos (120).

O *gap* percentual entre os melhores resultados do GA e os resultados obtidos através do método da organização variou entre 15,26% e 18,86%, evidenciando uma melhora significativa no consumo de matéria-prima ao aplicar o algoritmo proposto. Esse percentual representa, na prática, uma redução proporcional no volume de insumos necessários para atender à mesma demanda, implicando, entre outros aspectos, em menos sobras e na otimização na gestão de estoques. Como se trata de um insumo perecível que possui uma vida útil curta, essa diminuição no consumo previsto significa economia direta nos custos de aquisição e menor risco de perdas por vencimento. O que é particularmente importante para a redução de desperdícios, permitindo um planejamento de produção mais preciso.

Quanto ao tempo de execução médio nos experimentos, foi observado um tempo médio de 58,38 segundos considerando todas as instâncias, com maior tempo associado a instância 8, possivelmente devido a sua maior dimensão. Observa-se, ainda, que os resultados obtidos pelo algoritmo apresentam pequenas variações entre o melhor resultado e a média de cada instância, o que indica um comportamento consistente dos resultados nas diferentes execuções.

6. Considerações finais

Este trabalho propôs um Algoritmo Genético (GA) de objetivo único para resolução do problema de *Flow Shop Scheduling with Raw Waste Consumption Minimization* (FSS-RWC), em um ambiente de *flow shop* com múltiplas máquinas heterogêneas. A relevância do tema para o setor alimentício se justifica devido à natureza perecível dos insumos, exigindo decisões eficientes quanto ao sequenciamento de produção.

O algoritmo foi testado em oito instâncias de dados reais fornecidas pela empresa em estudo, apresentando desempenho superior ao método atualmente utilizado pela organização em todas as instâncias testadas, com *gaps* de melhoria variando entre 15,26% e 18,86% na redução do consumo de matéria-prima. Tais resultados foram obtidos em tempo computacional satisfatório levando em consideração a complexibilidade das instâncias em questão e suas particularidades.

Para trabalhos futuros, sugere-se a avaliação do algoritmo apresentado em instâncias de benchmark presentes na literatura, para avaliar sua eficácia. Com o mesmo objetivo pretende-se comparar as soluções do algoritmo genético com os resultados ótimos das instâncias da empresa, utilizando resolvedores exatos.

Referências

Anunay A., Pandey, A., & Krishna Kumar, S. (2023). Mathematical models for multi-stage hybrid assembly flow-shop scheduling with preventive maintenance and release times. **Computers and Industrial Engineering**, 186.

Aqil, S. (2024). Effective Population-Based Meta-heuristics with NEH and GRASP Heuristics Minimizing Total Weighted flow Time in No-Wait Flow Shop Scheduling Problem Under Sequence-Dependent Setup Time Constraint. **Arabian Journal for Science and Engineering**, 49(9), 12235–12258.

Chung, T. P., & Liao, C. J. (2013). An immunoglobulin-based artificial immune system for solving the hybrid flow shop problem. **Applied Soft Computing Journal**, 13(8), 3729–3736.

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br

Conserva, J. C. V., & Silva, Y. L. T. V. (2024). A hybrid evolutionary metaheuristic proposal applied to job-shop scheduling problems with earliness and tardiness penalties. **Revista Gestão & Tecnologia**, 24(1), 31-57.

Garavito-Hernández, E. A., Peña-Tibaduiza, E., Perez-Figueredo, L. E., & Moratto-Chimenty, E. (2019). A meta-heuristic based on the Imperialist Competitive Algorithm (ICA) for solving Hybrid Flow Shop (HFS) scheduling problem with unrelated parallel machines. **Journal of Industrial and Production Engineering**, 36(6), 362–370.

Goli, A.; Ala, A.; Hajiaghaei-keshteli, M. (2023). Efficient multi-objective meta-heuristic algorithms for energy-aware non-permutation flow-shop scheduling problem. **Expert Systems with Applications**, v. 213, p. 119077.

Guan, Y., Chen, Y., Gan, Z., Zou, Z., Ding, W., Zhang, H., & Ouyang, C. (2023). Hybrid flow-shop scheduling in collaborative manufacturing with a multi-crossover-operator genetic algorithm. **Journal of Industrial Information Integration**, 36, 100514.

Ilin, Vladimir et al. A hybrid genetic algorithm, list-based simulated annealing algorithm, and different heuristic algorithms for the travelling salesman problem. **Logic Journal of the IGPL**, v. 31, n. 4, p. 602-617, 2023.

Johnson, Selmer Martin. Optimal two-and three-stage production schedules with setup times included. **Naval research logistics quarterly**, v. 1, n. 1, p. 61-68, 1954.

PNUMA. *Relatório do Índice de Desperdício de Alimentos 2024*. Programa das Nações Unidas para o Meio Ambiente, 2024. Disponível em: https://www.unep.org/pt-br/resources/publicacoes/relatorio-do-indice-de-desperdicio-de-alimentos-2024

Sadki, H., Aqil, S., Belabid, J., & Allali, K. (2023). Multi-Objective Optimization Flow Shop Scheduling Problem Solving the Makespan and Total Flow Time with Sequence Independent Setup Time. **Journal of Advanced Manufacturing Systems.**

Silva, Y. L. T. V.; Subramanian, A.; Pessoa, A. A. (2018). Exact and heuristic algorithms for order acceptance and scheduling with sequence-dependent setup times. **Computers and Operations Research**, v. 90, p. 142–160.

Souza, M. A. A. Minimização do makespan em um sistema flexible flow shop scheduling utilizando métodos computacionais em uma organização beneficiadora de minérios. 2023.

Tangour, F.; Nouiri, M.; Abbou, R. Multi-objective production scheduling of perishable products in agri-food industry. **Applied sciences (Basel, Switzerland)**, v. 11, n. 15, p. 6962, 2021.

Yu, C., Semeraro, Q., & Matta, A. (2018). A genetic algorithm for the hybrid flow shop scheduling with unrelated machines and machine eligibility. **Computers and Operations Research**, 100.

Zhao, F., Bao, H., Wang, L., Xu, T., Zhu, N., & Jonrinaldi. (2022). A heuristic and meta-heuristic based on problem-specific knowledge for distributed blocking flow-shop scheduling problem with sequence-dependent setup times. **Engineering Applications of Artificial Intelligence,** 116.

Zhao, F.; Zhou, G.; Xu, T.; Zhu, N. (2023). A knowledge-driven cooperative scatter search algorithm with reinforcement learning for the distributed blocking flow shop scheduling problem. **Expert Systems with Applications**, v. 230, p. 120571.

DOI: 10.59254/sbpo-2025-212176
https://proceedings.science/p/212176?lang=pt-br