IEPG SUMMIT'25
Pensando o futuro com inteligência artificial e consciência social

DESENVOLVIMENTO DO ALGORITMO NEH EM *PYTHON* COM VISUALIZAÇÃO EM GRÁFICO DE GANTT PARA PROBLEMAS DE *FLOW SHOP*

Gustavo Queiroz Barberato – gubarberato@gmail.com
Alexandre Ferreira de Pinho – pinho@unifei.edu.br
Sandra Miranda Neves – sandraneves@unifei.edu.br

Palavras-chave: *Flow Shop Scheduling,* NEH (*Nawaz-Enscore-Ham*), Gráfico de Gantt, *Python*

1. INTRODUÇÃO

A otimização de sistemas produtivos é essencial para a competitividade e sustentabilidade das organizações na indústria moderna. Em ambientes produtivos complexos, o escalonamento da produção busca definir a sequência ideal de tarefas em máquinas, visando minimizar o *makespan*, ou seja, o tempo total de conclusão do processo (VIDOJEVIĆ *et al.*, 2025; KHURSHID *et al.*, 2025). Entre os modelos de escalonamento, o problema de *Flow Shop* destaca-se pela obrigatoriedade de todas as tarefas seguirem a mesma ordem de processamento nas máquinas. O desafio é encontrar a sequência que minimize o tempo ocioso e maximize a eficiência dos recursos.

Nesse contexto, o algoritmo NEH se destaca como uma das heurísticas mais eficazes e utilizadas para diminuir o *makespan* (NAWAZ; ENSCORE; HAM, 1983). Este trabalho tem como objetivo implementar o NEH em linguagem *Python*, aliado à visualização dos resultados por meio de Gráficos de Gantt, permitindo analisar e compreender o cronograma produtivo. A pesquisa, de caráter aplicado, exploratório e descritivo, visa desenvolver uma ferramenta prática e didática para auxiliar gestores e engenheiros na otimização do sequenciamento produtivo.

2. DESENVOLVIMENTO

2.1 Referencial teórico

2.1.1 O desafio do escalonamento em ambientes *Flow Shop*

O problema de *Flow Shop* de permutação busca determinar a melhor sequência de tarefas processadas em máquinas dispostas em série, de modo que o *makespan* — tempo total de conclusão das operações — seja o menor possível (VIDOJEVIĆ *et al.*, 2025; KHURSHID *et al.*, 2025). Por ser classificado como NP-difícil, a obtenção de soluções ótimas por métodos exatos torna-se inviável em instâncias de maior porte, o que motiva a aplicação de heurísticas como alternativa prática e eficiente.

2.1.2 Heurísticas de solução e a visualização de cronogramas

Entre essas abordagens, o algoritmo NEH destaca-se por sua simplicidade e elevado desempenho, sendo amplamente reconhecido como um dos métodos construtivos mais eficazes na minimização do *makespan* (NAWAZ; ENSCORE; HAM, 1983). Seu procedimento baseia-se na ordenação das tarefas de acordo com o tempo total de processamento e na inserção iterativa das mesmas na posição que resulta no menor tempo global de produção.

Figura 1 - Exemplo do algoritmo NEH.

	M1	M2	TOTAL
J1	4	2	4+2=6
J2	3	4	3+4=7
J3	5	9	5+9=14

J2-J3 = 17	M1	M2
J2	0+3=3	3+4=7
J3	3+5=8	8+9=17

J3-J2 = 18	M1	M2
J3	0+5=5	5+9=14
J2	5+3=8	14+4=18

Fonte: Os autores (2025).

Figura 2 - Minimização do *makespan*.

J1-J2-J3 = 21	M1	M2
J1	0+4=4	4+2=6
J2	4+3=7	7+4=11
J3	7+5=12	12+9=21

J2-J1-J3 = 21	M1	M2
J2	0+3=3	3+4=7
J1	3+4=7	7+2=9
J3	7+5=12	12+9=21

J2-J3-J1 = 19	M1	M2
J2	0+3=3	3+4=7
J3	3+5=8	8+9=17
J1	8+4=12	17+2=19

Fonte: Os autores (2025).

A Figura acima ilustra a aplicação do algoritmo NEH para a minimização do *makespan* em um problema de *Flow Shop*. O processo de construção da sequência ideal é demonstrado em etapas:

2.1.2.1 Cálculo dos tempos de processamento

Inicialmente, o tempo total de processamento para cada tarefa (J1, J2 e J3) é calculado pela soma dos tempos individuais em todas as máquinas (M1 e M2). A partir desses valores, as tarefas são ordenadas de forma decrescente com base nesse tempo total.

2.1.2.2 Construção da sequência parcial

As duas tarefas com os maiores tempos de processamento (J2 e J3) são selecionadas para iniciar o sequenciamento. As duas permutações possíveis (J2-J3 e J3-J2) são testadas para determinar qual delas resulta no menor *makespan* (tempo total de produção).

2.1.2.3 Inclusão da tarefa restante

A permutação com o menor *makespan* (J2-J3, com um valor de 17) é adotada. A tarefa restante (J1) é inserida em todas as posições possíveis da sequência parcial (início, meio e fim).

2.1.2.4 Seleção final

O *makespan* é recalculado para cada uma das novas permutações. A sequência que resultar no menor valor de produção é selecionada como a solução final do algoritmo, indicando a ordem ideal de produção para as três tarefas. Nesse caso, o sequenciamento ideia é J2-J3-J1, com um valor de 19.

2.1.3 Linguagem de programação para implementação do algoritmo e do Gráfico de Gantt

A implementação em *Python* é adequada devido à sua popularidade crescente na comunidade acadêmica e de engenharia como atestado por pesquisas recentes na área de escalonamento (Du; Zhou, 2025). Ademais, a disponibilidade de bibliotecas que permitem

cálculos otimizados (*NumPy*) e visualizações interativas por meio de Gráficos de Gantt auxiliam na decisão de utilizar essa linguagem de programação.

2.2 Metodologia

A metodologia adotada para o desenvolvimento do algoritmo NEH em *Python* caracteriza-se como uma pesquisa de natureza aplicada, com foco na criação de uma solução prática e replicável para o Problema de *Flow Shop* de Permutação (PFSP), visando a minimização do *makespan*.

O estudo combina os rigores da abordagem quantitativa – através da medição objetiva do desempenho do algoritmo e da manipulação de dados numéricos (tempos e sequências) – com a interpretação qualitativa fornecida pela análise visual do Gráfico de Gantt, que auxilia na identificação de ociosidades e gargalos no cronograma produtivo. O objeto central é o PFSP, um desafio de agendamento onde um conjunto de tarefas é processado sequencialmente em máquinas com o objetivo de encontrar a sequência ideal.

O processo de desenvolvimento foi dividido em quatro etapas:

2.2.1 Revisão bibliográfica e estruturação dos dados de teste

Consolidou-se o referencial teórico e foram estruturados conjuntos de dados fictícios e representativos, oriundos de *benchmarks* reconhecidos na literatura de escalonamento, garantindo a validação da eficácia do algoritmo em cenários padronizados.

2.2.2 Implementação em *Python*

Esta etapa prática foi subdividida em quatro módulos principais. O principal é a modelagem, que estabelece a inclusão do modelo de dados. O segundo é o Módulo de Implementação do Algoritmo, responsável pela tradução da lógica do algoritmo e do cálculo de *makespan*. Após isso, tem-se o Módulo de Avaliação e Refinamento, o qual faz uma comparação da sequência obtida com a original e inversa. Por fim, é realizado o Módulo de Interface com o Operador e Visualização, que gera o Gráfico de Gantt, oferecendo uma análise clara dos resultados.

2.2.3 Validação

A ferramenta foi validada em duas frentes: verificação manual em problemas de pequena escala e aplicação em conjuntos de dados mais complexos. Essa estrutura permite que a ferramenta esteja pronta para processar dados reais de um ambiente de produção posteriormente.

3. RESULTADOS E DISCUSSÃO

Os resultados esperados deverão comprovar a eficácia da heurística NEH na otimização de agendamentos de *Flow Shop*. Espera-se demonstrar que, ao aplicar o algoritmo a conjuntos de dados de *benchmark*, a sequência otimizada encontrada resultará consistentemente em um tempo total de conclusão inferior às sequências de processamento básicas (original ou invertida).

Uma das frentes do processo de validação já foi verificada, a qual apresenta a aplicação em problemas de pequena escala, utilizando os mesmos valores das figuras 1 e 2. É possível observar abaixo a comparação do sequenciamento definido pelo algoritmo e das sequências consideradas básicas, além da representação em Gráfico de Gantt.

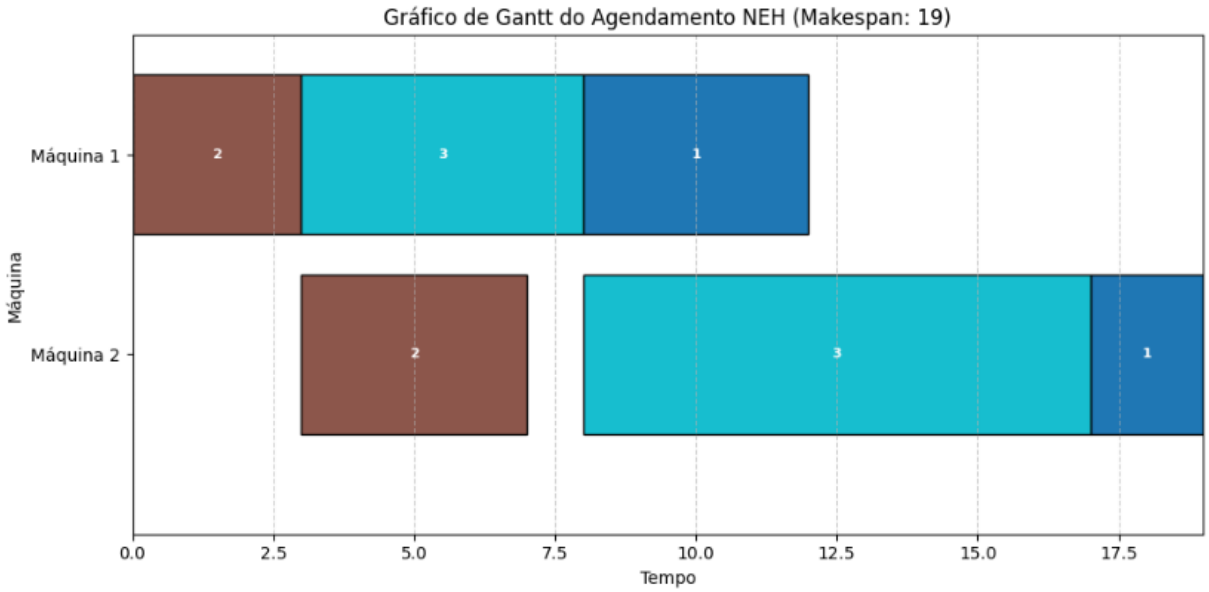
Figura 3 – Resultado em *Python* em um Problema de Pequena Escala.

```
Insira o número de tarefas: 3
Insira o número de máquinas: 2

Insira os tempos de processamento para 3 tarefas em 2 máquinas:
Tarefa 1:
  Tempo na Máquina 1: 4
  Tempo na Máquina 2: 2
Tarefa 2:
  Tempo na Máquina 1: 3
  Tempo na Máquina 2: 4
Tarefa 3:
  Tempo na Máquina 1: 5
  Tempo na Máquina 2: 9

--- Comparação com Sequências Simples ---
Makespan para ordem original: 21
Makespan para ordem invertida: 20

--- Resultados do Agendamento NEH ---
Agendamento Otimizado (IDs Originais das Tarefas):
[2, 3, 1]
Makespan Mínimo encontrado pela NEH: 19
```



Fonte: Os Autores (2025).

Após a verificação na primeira frente de validação, os próximos resultados serão obtidos através da aplicação da ferramenta em *benchmarks* mais robustos, onde será avaliada a performance do algoritmo e a eficácia da otimização em cenários de *Flow Shop* mais complexos.

4. CONSIDERAÇÕES FINAIS

O presente trabalho teve como objetivo principal implementar o Algoritmo NEH em linguagem *Python*, integrando-o à visualização dos resultados por meio de Gráficos de

Gantt, visando o desenvolvimento de uma ferramenta capaz de otimizar os Problemas de *Flow Shop* de Permutação (PFSP).

Na etapa da Implementação em *Python*, foram criados módulos dedicados à modelagem dos dados, à tradução lógica do NEH e, por fim, ao módulo de interface e visualização que permite a análise do cronograma produtivo.

Os resultados preliminares da pesquisa atestam o sucesso da validação em pequena escala. Conforme evidenciado na seção 3, a aplicação do algoritmo a problemas simplificados (utilizando os mesmos dados de referência das figuras 1 e 2) demonstrou a validação do código.

Embora esta fase do estudo ateste a validade técnica da implementação, a conclusão completa do trabalho depende dos resultados esperados na próxima etapa de validação. O resultado esperado é que o algoritmo NEH demonstre uma redução do *makespan* em relação aos métodos básicos, provando a eficácia da heurística. Ademais, espera-se que o Gráfico de Gantt facilite o processo de identificação de ociosidades e gargalos na linha de produção.

AGRADECIMENTOS

Os autores agradecem ao Programa de Educação Tutorial (PET) do curso de Engenharia de Produção da Universidade Federal de Itajubá (UNIFEI). Um agradecimento especial à Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG) pelo apoio.

REFERÊNCIAS

DU, Xinzhe; ZHOU, Yanping. **A Novel Hybrid Differential Evolutionary Algorithm for Solving Multi-objective Distributed Permutation Flow-Shop Scheduling Problem**. International Journal of Computer Intelligence Systems, [*s. l.*], v. 18, n. 1, p. 1–22, 2025.

KHURSHID, Bilal *et al.* **A hybrid evolution strategies algorithm for non-permutation flow shop scheduling problems**. Scientific Reports, [*s. l.*], v. 15, n. 1, p. 1–21, 2025.

NAWAZ, Muhammad; ENSCORE, Edward E.; HAM, I. **A Heuristic Algorithm for the m-Machine, n-Job Flow-shop Sequencing Problem**. Omega, [*s. l.*], v. 11, n. 1, p. 91–95, 1983.

VIDOJEVIĆ, Filip *et al.* **A novel artificial intelligence search algorithm and mathematical model for the hybrid flow shop scheduling problem**. Journal of Big Data, [*s. l.*], v. 12, n. 1, p. 1–24, 2025.