# [UFS 2024.2] Resolução do Problema de Seleção Ótima de Pedidos em Waves - SBPO 2025

**Aluno:** José Wilson Martins Filho

**Matrícula:** 201900051006

O presente trabalho explica e demonstra uma solução para a prova da Unidade 2 da disciplina de Inteligência Artificial (2024.2). A prova se baseia no Desafio SBPO 2025 - Problema da Seleção de pedidos Ótima.

A solução desse projeto também pode ser lida em no repositório no github.

---

## Parte 1: Modelagem do Problema como CSP

**Definição dos Componentes (X, D, C):**

- **Variáveis (X):**
  - **Pedidos:** O0, O1, O2, O3, O4
  - **Corredores:** A0, A1, A2, A3, A4
- **Domínios (D):**
  Cada variável pode assumir o valor 0 (não selecionado) ou 1 (selecionado).
- **Restrições (C):**
  1. **Total de Unidades:**
     Se todas as variáveis dos pedidos estiverem atribuídas, a soma das unidades dos pedidos selecionados deve estar entre o limite inferior (LB) e o limite superior (UB).
  2. **Disponibilidade:**
     Quando todas as variáveis (pedidos e corredores) estão atribuídas, para cada item a quantidade total solicitada (nos pedidos selecionados) não pode exceder a quantidade disponível (nos corredores selecionados).

**Função Objetivo:**
A produtividade é definida pela média de itens coletados por corredor, isto é:

```
Objetivo = Total de Unidades dos Pedidos Selecionados / Número de Corredores Selecionados
```

Para desempate, também se considera o total de unidades – ou seja, entre soluções com a mesma média, a que tiver maior total de unidades é considerada melhor.

---

## Parte 2: Implementação da Solução com AIMA-Python

O código completo da solução pode ser encontrado em aima/unidade2_mercadolivre.py.

Esta implementação utiliza as estruturas de dados básicas para representar o problema:

- **Dicionários para os dados:**
  Os dados dos pedidos e dos corredores são armazenados em dicionários (`orders_data` e `corridors_data`), onde cada chave representa um identificador e o valor é uma lista com as quantidades dos itens.
- **Listas para as variáveis:**
  As variáveis do problema são definidas como a união dos identificadores de pedidos e de corredores.
- **Dicionários para os domínios e vizinhança:**
  Cada variável tem como domínio o conjunto {0, 1} e as estruturas `domains` e `neighbors` são definidas para facilitar a aplicação das restrições.

A classe **OrderCSP** foi construída estendendo a classe base **CSP** do AIMA, que serviu de fundamento para nossa solução. Ela redefine métodos cruciais, como:

- `nconflicts` – que verifica os conflitos globais, utilizando a função `constraints` para validar a consistência de uma atribuição.
- `assign` e `unassign` – para gerenciar as atribuições das variáveis.
- `select_unassigned_variable` – para escolher a próxima variável a ser atribuída durante a busca.

A função `all_solutions` implementa uma busca por backtracking que enumera todas as soluções viáveis. Para cada solução, é calculado o valor objetivo (um par contendo a média de itens por corredor e o total de unidades), o que permite comparar diferentes waves geradas e, finalmente, selecionar a solução que maximiza a produtividade.

Os passos para rodar a solução são:

1. Clone o projeto:
   ```
   $ git clone https://github.com/wilfilho/aima-python
   ```
2. Entre no diretório:
   ```
   $ cd aima-python
   ```
3. Crie um ambiente virtual:
   ```
   $ python3 -m venv csp-env
   ```
4. Ative o ambiente:
   ```
   $ source csp-env/bin/activate
   ```
5. Instale os pacotes necessários:
   ```
   $ pip install -r aima/requirements.txt
   ```
6. Rode o código:
   ```
   $ python aima/unidade2_mercadolivre.py
   ```

---

## Parte 3: Testes, Validação e Análise de Desempenho

### Teste com o Exemplo Fornecido

**Dados do Exemplo:**

- **Pedidos:**
  - O0: [3, 0, 1, 0, 0]

    - O1: [0, 1, 0, 1, 0]
    - O2: [0, 0, 1, 0, 2]
    - O3: [1, 0, 2, 1, 1]
    - O4: [0, 1, 0, 0, 0]

- **Corredores:**

    - A0: [2, 1, 1, 0, 1]
    - A1: [2, 1, 2, 0, 1]
    - A2: [0, 2, 0, 1, 2]
    - A3: [2, 1, 0, 1, 1]
    - A4: [0, 1, 2, 1, 2]

- **Limites:** LB = 5 e UB = 12

**Cálculo do Valor Objetivo:**
Cada wave é avaliada pela função objetivo, que calcula:

```
Média = (Total de Unidades dos Pedidos Selecionados) / (Número de Corredores Selecionados)
```

Por exemplo, se a solução seleciona os pedidos ['O0', 'O1', 'O2', 'O4'] com um total de 10 unidades e os corredores ['A1', 'A3'] (2 corredores), então:

```
Média = 10 / 2 = 5
```

Essa média, juntamente com o total de unidades, forma o par objetivo usado para comparar as soluções.

**Resultado Obtido:**

A saída do algoritmo lista todas as soluções encontradas, por exemplo:

- *Solução 1:* Pedidos ['O3'], Corredores ['A3', 'A4'], Objetivo (média, total) = (2.5, 5)
- *Solução 2:* Pedidos ['O3'], Corredores ['A2', 'A3', 'A4'], Objetivo (média, total) = (1.67, 5)
- ...
- *Solução 309:* Pedidos ['O0', 'O1', 'O2', 'O4'], Corredores ['A0', 'A1', 'A2', 'A3', 'A4'], Objetivo (média, total) = (2.0, 10)

A solução ótima escolhida pelo algoritmo foi:

```
==================== SOLUÇÃO ÓTIMA ====================
Pedidos selecionados: ['O0', 'O1', 'O2', 'O4']
Corredores selecionados: ['A1', 'A3']
Número de corredores selecionados: 2
Total de unidades dos pedidos selecionados: 10
Valor objetivo (média de itens por corredor): 5.0
Tempo de execução: 0.0179 segundos
=======================================================
```

A solução proposta retornou o **resultado esperado**.

## Comparação de Waves Geradas

**Solução 1**

- **Pedidos:** ['O3']
- **Corredores:** ['A3', 'A4']
- **Objetivo:** (média = 2.5, total = 5)

Nesta solução, apenas o pedido O3 é selecionado, resultando em 5 unidades no total. Como são usados 2 corredores, a média é baixa (2.5), indicando baixa eficiência.

**Solução 41**

- **Pedidos:** ['O2', 'O3']
- **Corredores:** ['A1', 'A4']
- **Objetivo:** (média = 4.0, total = 8)

Aqui, os pedidos O2 e O3 totalizam 8 unidades. Com 2 corredores, a média é de 4.0, melhorando a eficiência, mas ainda abaixo da solução ótima.

**Solução 21**

- **Pedidos:** ['O3', 'O4']
- **Corredores:** ['A3', 'A4']
- **Objetivo:** (média = 3.0, total = 6)

Nesta configuração, os pedidos O3 e O4 somam 6 unidades, resultando em uma média de 3.0 com 2 corredores. Embora melhor que a solução com um único pedido, ainda não maximiza a produtividade.

**Solução Ótima**

- **Pedidos Selecionados:** ['O0', 'O1', 'O2', 'O4']
- **Corredores Selecionados:** ['A1', 'A3']
- **Objetivo:** (média = 5.0, total = 10)

A solução ótima agrupa quatro pedidos, atingindo 10 unidades. Com 2 corredores, a média é de 5.0, evidenciando um equilíbrio ideal entre quantidade e uso de recursos.

**Detalhes**

- **Cobertura dos Pedidos:**
  A solução ótima inclui 4 pedidos, enquanto as demais englobam apenas 1 ou 2. Mais pedidos resultam em maior total de unidades e, consequentemente, melhor média.

- **Utilização dos Corredores:**
  Apesar de todas as soluções utilizarem 2 corredores, a distribuição dos pedidos na solução ótima maximiza a eficiência, aumentando a média de itens por corredor.

- **Valor Objetivo:**
  A solução ótima apresenta o maior valor (5.0) quando comparada aos valores 2.5, 3.0 e 4.0 das

outras soluções, demonstrando a importância de agrupar pedidos de forma estratégica para maximizar a produtividade.

### Análise de Desempenho do CSP

- **Cenário Atual:**
  No exemplo atual, com 10 variáveis e 1024 combinações possíveis, a enumeração completa via backtracking se mostrou extremamente eficiente, com um tempo de execução de apenas 0.0179 segundos. Esse resultado evidencia que, para o problema em escala atual, o algoritmo é muito rápido e eficaz.

  No entanto, é importante notar que o tempo de execução está diretamente relacionado ao tamanho do espaço de busca. Se o número de variáveis aumentar, o tempo de execução pode crescer exponencialmente, o que ressalta a importância de otimizações e heurísticas para problemas maiores.

  - **Cenários de Maior Escala:**
    Em problemas com um número maior de variáveis (por exemplo, com muitos pedidos e/ou muitos corredores), o espaço de busca cresce exponencialmente. Para esses casos, recomenda-se:
  - Utilizar heurísticas como MRV (Minimum Remaining Values) para seleção de variáveis.
  - Aplicar LCV (Least Constraining Value) para ordenar os valores.
  - Implementar técnicas de propagação de restrições.

- **Conclusão do Desempenho:**
  A abordagem atual é adequada para o exemplo fornecido. Para aplicações em larga escala, adaptações e otimizações serão necessárias para manter a eficiência da resolução.

---

## Conclusão

A solução desenvolvida foi capaz de encontrar a solução ótimo descrita pelo problema. Além disso, a solução modela o problema de seleção ótima de pedidos em waves como um CSP, definindo explicitamente as variáveis, os domínios e as restrições, e utiliza a implementação do AIMA como base. A solução ótima encontrada pelo algoritmo foi:

- **Pedidos Selecionados:** ['O0', 'O1', 'O2', 'O4']
- **Corredores Selecionados:** ['A1', 'A3']
- **Número de Corredores:** 2
- **Total de Unidades:** 10
- **Valor Objetivo (Média):** 5.0
- **Tempo de Execução:** 0.0179 segundos

O tempo de execução exibido, demonstrando que, embora a enumeração completa seja viável para o exemplo atual, métodos heurísticos serão essenciais para problemas de maior escala.