O problema clássico do transporte em Pesquisa Operacional: expansão em 3 fatores (2023 – 2º semestre)

Filipe Augusto Marques de Paula e Filipe Santos Fernandes
Universidade Federal de Minas Gerais - UFMG

Abstrato – Abordamos o conhecido problema de transporte, transbordo e designação do campo de Pesquisa Operacional. Problemas clássicos incluem e tratam esse tipo de problema em termos didáticos, em geral, com dois fatores. Aqui, o desafio foi o de incluir um terceiro elemento impactante tanto em custos quanto em restrições. Obviamente, como será citado, existem abordagens dentro desta classe bem mais complexas envolvendo problemas reais, mas a proposta foi a de equilibrar o nível didático deste problema com o aumento de complexidade.

Palavras-chave: demanda, logística, oferta, otimização.

I. Introdução

Este problema faz parte de uma classe de problemas de otimização combinatória com diversas aplicações incluindo: logística, distribuição, alocação de recursos e transporte. A principal característica deste tipo de desafio envolve a melhor maneira de mover bens de múltiplas origens a múltiplos destinos, considerando restrições de capacidade da fonte, restrições de demanda dos consumidores, custos, etc.

A motivação dos estudos por trás desse tipo de problema surge, inicialmente, na década de 40 e, inclusive, esse problema foi um dos primeiros problemas de programação linear a serem formalmente estudados[1]. O problema de transporte lida com a movimentação de mercadorias de várias fontes para vários destinos e, como adição de complexidade, incluímos a diferenciação dos produtos. Com isso, além de vários centros de origem, vários centros de destino, temos, nesse estudo, vários produtos diferentes.

Os livros de Pesquisa Operacional lidam (geralmente), em sua modelagem, com dois elementos preponderantes representando os centros de distribuição de produtos e os mercados consumidores. Neste artigo, basearemos nossos resultados considerando um novo termo: o tipo de produto.

II. Revisão da literatura

O objetivo desse problema de otimização é minimizar os custos de transporte considerando a oferta dos centros de produção, a demanda dos mercadores consumidores, o custo de se transportar determinado tipo de produto de uma origem para um destino e dentre outros. Em geral, é admitida a hipótese de que são conhecidas as quantidades ofertadas e demandada dos produtos; admite-se também que o peso de se transportar determinado tipo de produto é conhecido (mesmo que com certa abstração).

Vamos relembrar a formulação matemática para esse tipo de problema. (Nota: a formulação inicial será feita sem o terceiro fator que incluímos para que possamos fazer a analogia posteriormente) Inicialmente, iremos assumir que existam m centros de origem, n mercados consumidores e que o custo para se transportar uma unidade do produto da origem i para o destino j é c_{ij}. Definimos também que a oferta do produto na origem i é a_i e a demanda no destino j é b_j. Como variáveis, definimos x_{ij} a quantidade transportada do produto da origem i para o destino j em que essas variáveis são "não negativas". Se x_{ij} é a quantidade transportada, então $c_{ij}x_{ij}$ é o custo para se realizar esse transporte. O custo total é dado pela soma de todos os custos de transporte de todos os centros de origem para todos os mercados consumidores. Logo temos:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

Esse é o custo que deve ser minimizado. Um olhar sobre as restrições mostra que aquilo que é transportado de cada origem não pode ultrapassar a quantidade disponível em cada centro de origem *i.* Em outros termos, tem-se:

$$\sum_{j=1}^{n} x_{ij} \leq a_i \quad \forall i \in \{1,2 \dots, m\}$$

As demandas dos mercados consumidores também precisam ser atendidas. Por isso, definimos uma outra restrição para este caso em que:

$$\sum_{i=1}^{m} x_{ij} = b_j \quad \forall j \in \{1,2 \dots, n\}$$

[1] O trabalho de George Dantzig na resolução de problemas de programação linear, especialmente o algoritmo simplex, contribuiu de forma extremamente significativa para a compreensão e resolução desses problemas.

[2] O artigo *The transportation problem with packing constraints* foi escrito pelos autores Tülay Flamand, Manuel Iori, Mohamed Haouari e publicado em setembro de 2023 em *Computers & Operations Research – Volume 157.*

[3] O artigo *A location–transportation problem under demand uncertainty for a pharmaceutical network in Brazil* foi elaborado pelos autores Aura Jalal, Eli Angela Vitor Toso e Reinaldo Morabito e publicado em junho de 2023 em *Computers & Chemical Engineering – Volume 174.*

A modelagem completa fica então da seguinte forma:

$$\min f(x_{11}, x_{12}, \dots x_{mn}) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$sujeito\ a: \begin{cases} \sum_{j=1}^{n} x_{ij} \le a_i \quad \forall i \in \{1,2 \dots, m\} \\ \sum_{i=1}^{m} x_{ij} = b_j \quad \forall j \in \{1,2 \dots, n\} \end{cases}$$

$$x_{ij} \ge 0\ \forall i \in \{1,2 \dots, m\}\ e\ \forall j \in \{1,2, \dots, n\}$$

No artigo denominado *The transportation problem with packing constraints* [2], esse mesmo problema foi abordado com algumas variações interessantes. Foram incluídas restrições de empacotamento associadas às capacidades dos veículos responsáveis pelo transporte. Além disso, o custo total é calculado levando em conta o custo de aquisição das mercadorias e o custo fixo do transporte de forma que a aquisição de mercadorias é diferente para cada fornecedor. Para cada relação entre origem e destino, o custo fixo do transporte depende da capacidade do veículo bem como da distância entre os locais. Em resumo, o problema consiste na distribuição das mercadorias de um ponto de origem, na seleção de um tipo específico de veículo e na especificação de embalagens viáveis para cada veículo selecionado. O problema foi resolvido como um PLIM (Programação Linear Inteira Mista) e baseado numa abordagem de busca por vizinhança variável. Os autores afirmam que usaram uma heurística de decomposição que fornecem uma solução inicial viável para a busca por vizinhança variável e utilizaram em sua abordagem a plataforma CPLEX e o GUROBI. Em ambas as plataformas, encontraram resultados viáveis (CPLEX encontrando as soluções em uma média de 0,4 a 22,4s e GUROBI em uma média de 0,5 a 91,8s).

Em um outro artigo denominado *A location–transportation problem under demand uncertainty for a pharmaceutical network in Brazil* [3], os autores trabalharam com a "incerteza" da demanda dos mercados consumidores, isto é, demandas que são variáveis por diversos motivos como incertezas estratégias (mudanças no contexto sociopolítico) e operacionais (mudanças nas operações da cadeia de abastecimento). Os custos fixos de transporte também sofrem alteração devido ao alto valor agregado dos produtos que estão sendo transportados e, por este motivo, existe a contratação de seguradoras que aumentam o custo fixo de transporte para esse caso. Além disso, as mesmas seguradoras impõem restrições sobre os limites monetários ao transporte de cargas, ou seja, o valor que está sendo transportado tem um limite que ainda é coberto pela seguradora contra roubos, perdas ou quaisquer outros problemas que podem afetar o envio. Isso ainda sem contar restrições que são impostas pelo próprio governo federal como o imposto de ICMS (Imposto sobre circulação de mercadorias e serviços). Foi utilizada a heurística *Fix and Optimize* para a resolução desse problema com o seguinte algoritmo:

Algorithm 1: Fix-and-Optimize - *F&O4*

```
Initialization: Initial solution ;
Fix the variables larger than zero in their current values ;
Incumbent solution := initial solution, OF_incumbent := objective function of the initial solution, OF_MIP := objective function of the subproblem ;
for t = 1 to |T| do
    Unfix the discrete variables Z_ijkt ;
    Solve the resulting subproblem ;
    if OF_MIP < OF_incumbent then
        Incumbent solution := MIP solution ;
        OF_incumbent:= OF_MIP ;
        Fix variables larger than zero according to the incumbent solution.
    end
    Unfix the discrete variables Z'_jkt ;
    Solve the resulting subproblem ;
    if OF_MIP < OF_incumbent then
        Incumbent solution := MIP solution ;
        OF incumbent:= OF MIP ;
        Fix variables larger than zero according to the incumbent solution.
    end
end
```

Fig. 1: O algoritmo fix and optimize usado para resolver o problema.

Os autores fizeram uso do conjunto de dados baseado nas operações atuais de uma empresa farmacêutica no Brasil. Para incorporar a incerteza das instâncias (pelos motivos já citados anteriormente), eles definiram desvios de demanda. Em resumo, o modelo considera características reais, como estruturas tributárias, segurança no transporte de carga, múltiplas alternativas de transporte, múltiplos produtos com características diferentes (refrigerados e não refrigerados, p.e.) e incerteza de demanda.

III. DESCRIÇÃO DO PROBLEMA E MODELAGEM

Agora, passaremos a discutir a proposta desta pesquisa de forma integral. O objetivo desta pesquisa não foi o de representar um caso estritamente real com todas as suas complexidades específicas como os vistos anteriormente, pois (i) um problema real pode demorar meses para ser modelado e totalmente coberto e (ii) o alvo é ser introduzido aos softwares e ferramentas utilizados no campo de Pesquisa Operacional. Com isso, trabalhamos no âmbito de superar a complexidade dos problemas definidos nos livros didáticos e, ao mesmo tempo, manter a curva de aprendizagem sem introdução de conceitos demasiadamente avançados.

Portanto incluíremos um novo parâmetro de análise neste modelo e o denominaremos de k que representa o tipo de produto que está sendo transportado num total de p produtos. As variáveis, antes denominadas x_{ij} agora passarão a ser denominadas x_{ijk} e o custo de se transportar determinado produto k de um ponto de origem i para um destino j é c_{ijk} . O custo total então é refeito considerando essa nova variável:

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} c_{ijk}x_{ijk}$$

As restrições de oferta e demanda também são alteradas considerando o novo fator k. Portanto, o modelo que abordaremos nesta pesquisa é o seguinte:

$$\min f(x_{111}, x_{112}, \dots x_{mnp}) = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} c_{ijk}x_{ijk}$$

$$sujeito\ a: \begin{cases} \sum_{k=1}^{p}\sum_{j=1}^{n} x_{ijk} \le a_{ik} \quad \forall i \in \{1,2 \dots, m\} \\ \sum_{k=1}^{p}\sum_{i=1}^{m} x_{ijk} = b_{jk} \quad \forall j \in \{1,2 \dots, n\} \end{cases}$$

$$x_{ijk} \ge 0 \in \mathbb{Z}_+$$

$$\forall i \in \{1,2 \dots, m\}, \forall j \in \{1,2, \dots, n\}\ e\ \forall k \in \{1,2, \dots, p\}$$

As tabelas abaixam ilustram algumas instâncias de ofertantes, demandantes e produtos para melhor compreensão. Ao longo da pesquisa, variamos os tamanhos das instâncias para análise do modelo e verificação de comportamento.

TABELA I
DISTÂNCIAS ENTRE ORIGEM E DESTINO

Origem	Belo Horizonte	Porto Alegre	Recife	Fortaleza	Curitiba	Manaus	Belém	Goiânia	Contagem
São Paulo	586	1.119	2.196	2.331	336	2.804	2.482	923	680
Rio de Janeiro	339	1.532	2.071	2.500	853	3.082	2.803	924	440
Brasília	716	1.817	1.652	2.135	1.319	2.219	1.643	173	750
Rio Grande do Sul	1.000	300	2.900	3.500	1.800	4.000	3.200	800	950
Salvador	1.336	2.686	796	1.342	2.409	2.916	2.466	1.591	1.280

As distâncias entre as cidades foram baseadas em kilômetros. Temos também a representação dos tipos de produtos que simulamos e seus respectivos "pesos" ou "custos" de transporte:

TABELA II
TIPOS DE PRODUTOS E SEUS CUSTOS

Produto	Taxa de transporte
Arroz	3
Feijão	3
Óleo	5
Carne	6

Analogamente, temos a representação das ofertas e demandas deste problema:

TABELA III
OFERTAS DOS CENTROS DE DISTRIBUIÇÃO PARA CADA PRODUTO

Distribuidora	Arroz(kg)	Feijão (kg)	Óleo (L)	Carne
São Paulo	5.000	3000	6.000	3000
Rio de Janeiro	4.500	2500	5.500	2500
Brasília	4.000	2700	5.200	6000
Rio Grande do Sul	500	600	800	8000
Salvador	4.300	2800	5.800	1500
Total oferta	**18300**	**11600**	**23300**	**21000**

TABELA IV
DEMANDAS DOS MERCADOS CONSUMIDORES PARA CADA PRODUTO

Cidade Consumidora	Arroz (kg)	Feijão (kg)	Óleo (L)	Carne
Belo Horizonte	3.800	2.600	5.200	3.500
Porto Alegre	2.375	1.500	2.100	4.700
Recife	1.900	1.000	2.300	2.800
Fortaleza	2.500	1.800	3.400	2.800
Curitiba	1.750	1.050	2.150	2.500
Manaus	1.100	1.050	1.500	3.050
Belém	1.650	950	1.650	300
Goiânia	1.650	1.050	1.750	500
Contagem	800	450	1.000	800
Total demanda	**17.525**	**11.450**	**21.050**	**20.950**

IV. DESCRIÇÃO DO ALGORITMO

Na resolução desse problema, utilizamos o pacote comercial GUROBI em conjunto com a linguagem de programação Pyhton que contém a biblioteca *Gurobipy*. Para obtenção dos valores (destacados nas tabelas acima) utilizamos a biblioteca *pandas* que também faz parte do Python. Abaixo está o algoritmo que foi utilizado tanto para tratamento dos dados quanto para criação do modelo:

ALGORITMO - TRATAMENTO DE DADOS

```
#Importação de bibliotecas
 Importar a biblioteca Gurobi como gp
 Importar a biblioteca Pandas como pd

#Leitura do arquivo Excel
 Abrir o arquivo 'Base_de_dados_PO.xlsx'
 Ler dados das planilhas 'Distâncias', 'Pesos k', 'Oferta' e  'Demanda b(j,k)'

 #Obtenção do tamanho dos conjuntos
Determinar o número de distribuidoras (I)
Determinar o número de clientes (J)
Determinar o número de produtos (K)

#Criação de listas para rótulos
Criar lista 'distr' contendo as distribuidoras
Criar lista 'client' contendo os clientes
Criar lista 'prod' contendo os produtos

#Criação de dicionários para ofertas, demandas e custos
Criar dicionário 'a' para guardar valores das ofertas por distribuidora e produto
Criar dicionário 'b' para guardar valores das demandas por cliente e produto
Criar dicionário 'custos' para armazenar os custos de transporte entre
distribuidoras, clientes e produtos
```

Fig. 2. Pseudo-código para aquisição dos dados e tratamento

Dessa forma, todo o trabalho agora será processado pelo python que, através da biblioteca *gurobipy*, preparará os dados para que possa ser criado o modelo de acordo o proposto anteriormente. Abaixo temos o pseudo-código para criação do modelo de otimização:

ALGORITMO - CRIAÇÃO DO MODELO DE OTIMIZAÇÃO

```
#Criação do modelo de otimização
Criar um modelo de otimização usando Gurobi;

#Definição das variáveis de decisão
Para cada distribuidora i em distr:
  Para cada cliente j em client:
    Para cada produto k em prod:
      Adicionar uma variável de decisão x[i, j, k] ao modelo

#Definição da função objetivo
Para cada distribuidora i em distr:
  Para cada cliente j em client:
    Para cada produto k em prod:
      Adicionar o termo de custo x[i, j, k] * custos[i, j, k] à função objetivo.

#Definição das restrições de factibilidade
Verificar a factibilidade para cada produto k em prod:
  Calcula soma total de ofertas para o produto k (soma_ofertas)
  Calcula soma total de demandas para o produto k (soma_demandas)
  Se soma_ofertas > soma_demandas:
    Definir factibilidade[k] como 1
  Senão:
    Definir factibilidade[k] como 0

#Adição das restrições de oferta ao modelo
Para cada distribuidora i em distr:
  Para cada produto k em prod:
    Se factibilidade[k] for verdadeiro:
      Adicionar a restrição x[i, j, k] <= a[i, k] para cada cliente j em client
    Senão:
      Soma das qtds de k enviadas por i p/ j é igual à oferta de i para k

#Adição das restrições de demanda ao modelo
Para cada cliente j em client:
  Para cada produto k em prod:
    Se factibilidade[k] for verdadeiro:
      Adicionar a restrição x[i, j, k] == b[j, k] para cada distribuidora i em distr
    Senão:
      Soma das qtds k recebidas por j de i é <= à demanda de j para k
```

Fig. 3. Pseudo-código para criação do modelo de otimização

A partir disso, executamos o modelo com o comando *(m.optimize())* sendo 'm' a variável que contém o nosso modelo criado.

É importante considerar uma adaptação importante que existe em caso de infactibilidade do modelo. Essa adaptação é feita nas restrições e pode ser verificado no algoritmo de criação do modelo nas linhas 31 e 39. Caso os centros de origem não consigam cobrir a demanda de um produto k, as restrições de demanda são atualizadas para <=, isto é, os mercadores consumidores podem receber um pouco abaixo da demanda permitida. Isso, apenas em caso de infactibilidade de restrições; é uma forma de tentar reduzir o custo mesmo com as restrições sendo violadas.

Por fim, armazenamos o "Plano de Transporte" em uma planilha para que o resultado possa ser melhor visualizado. Uma amostra do resultado pode ser visualizado na tabela abaixo:

TABELA V
VISUALIZAÇÃO DO PLANO DE TRANSPORTE APÓS OTIMIZAÇÃO

Produto	Origem	Belo Horizonte	Belém	Contagem	Curitiba	Fortaleza	Goiânia	Manaus	Porto Alegre	Recife	Soma
Arroz	Brasília	0	1650	0	0	0	1650	700	0	0	4000
	Rio Grande do Sul	0	0	0	0	0	0	0	500	0	500
	Rio de Janeiro	3800	0	700	0	0	0	0	0	0	4500
	Salvador	0	0	0	0	2400	0	0	0	1900	4300
	São Paulo	0	0	100	1750	100	0	400	1875	0	4225
	Total Arroz	3800	1650	800	1750	2500	1650	1100	2375	1900	
Carne	Brasília	0	300	0	0	1350	0	3050	0	1300	6000
	Rio Grande do Sul	1950	0	800	0	0	500	0	4700	0	7950
	Rio de Janeiro	1550	0	0	0	950	0	0	0	0	2500
	Salvador	0	0	0	0	0	0	0	0	1500	1500
	São Paulo	0	0	0	2500	500	0	0	0	0	3000
	Total Carne	3500	300	800	2500	2800	500	3050	4700	2800	
Feijão	Brasília	0	950	0	0	0	1050	700	0	0	2700
	Rio Grande do Sul	0	0	0	0	0	0	0	600	0	600
	Rio de Janeiro	2500	0	0	0	0	0	0	0	0	2500
	Salvador	0	0	0	0	1800	0	0	0	1000	2800
	São Paulo	100	0	450	1050	0	0	350	900	0	2850
	Total Feijão	2600	950	450	1050	1800	1050	1050	1500	1000	
Óleo	Brasília	0	1650	0	0	0	1750	1500	0	0	4900
	Rio Grande do Sul	0	0	0	0	0	0	0	800	0	800
	Rio de Janeiro	5200	0	300	0	0	0	0	0	0	5500
	Salvador	0	0	0	0	3400	0	0	0	2300	5700
	São Paulo	0	0	700	2150	0	0	0	1300	0	4150
	Total Óleo	5200	1650	1000	2150	3400	1750	1500	2100	2300	

V. RESULTADOS E ANÁLISE

O modelo foi resolvido em um *PC Microsoft com uma CPU 11th Gen Intel(R) Core(TM) i5-1145G7 @ 2.60GHz* através do software de otimização *Gurobi Optimizer version 10.0.3 build v10.0.3rc0*. O critério de parada foi a otimalidade do modelo (como o programa convergiu em tempo suficiente para as instâncias de teste, não definimos outro critério que não a otimalidade).

Algumas variações de instâncias foram feitas para teste do modelo de otimização. Vamos dividir em três tipos de testes:

1. *Teste de otimização geral;*
2. *Teste de adaptação para infactibilidade;*
3. *Teste com 'super-instância' e dados aleatórios.*

No *teste de otimização geral*, geramos poucas instâncias para verificar como o modelo respondia e também para adequar a visualização da resposta deste. Não forçamos a infactibilidade neste caso pois deixamos este teste em específico para o teste 2. A tabela de resposta daquele teste já foi, inclusive, mostrada anteriormente juntamente com as restrições de oferta e demanda. *Verifiquemos uma amostra para validarmos o modelo em que o mercado consumidor é Belo Horizonte e o produto consumido é a carne.* Na Tabela IV, a demanda por carne de Belo Horizonte é de 3.500; já na Tabela V, o modelo resolve esse modelo respeitando as restrições de demanda da Tabela IV e as restrições de oferta da Tabela III. Como resultado, ele faz um balanço de envio de produtos entre os ofertantes de carne e conclui que o envio será melhor otimizado se vir do estado de Rio Grande do Sul e da cidade do Rio de Janeiro.

No *teste de adaptação para infactiblidade*, aumentamos o número de instâncias e usamos o máximo possível das ofertas disponíveis para atender às demandas. Como este não é o objetivo primário, geramos um *log* de análise em que é informado quantas unidades faltaram e em quais locais conforme amostra abaixo:

```
Faltaram 129 unidades de Feijão para o cliente Consumidor 841
Faltaram 1254 unidades de Feijão para o cliente Consumidor 842
Faltaram 3491 unidades de Feijão para o cliente Consumidor 844
Faltaram 1422 unidades de Feijão para o cliente Consumidor 848
Faltaram 1924 unidades de Feijão para o cliente Consumidor 849
Faltaram 62 unidades de Feijão para o cliente Consumidor 859
Faltaram 1008 unidades de Feijão para o cliente Consumidor 873
Faltaram 1620 unidades de Feijão para o cliente Consumidor 886
Faltaram 483 unidades de Feijão para o cliente Consumidor 887
Faltaram 645 unidades de Feijão para o cliente Consumidor 889
Faltaram 41 unidades de Feijão para o cliente Consumidor 890
Faltaram 1749 unidades de Feijão para o cliente Consumidor 909
Faltaram 1535 unidades de Feijão para o cliente Consumidor 913
Faltaram 2096 unidades de Feijão para o cliente Consumidor 921
```

Fig. 4. *Flag* de falta de produtos para consumidores

Para o *teste com 'super-instância' e dados aleatórios*, a base de dados continha 2249 distribuidoras, 4 produtos e 2499 mercadores consumidores. O Gurobi identificou 18984 linhas, 22462016 colunas e 44924032 "não-zeros". Inicialmente o *pre-solve* realizou 18 tentativas de simplificação do problema e, na 18º iteração conseguiu remover 14238 linhas e 16846512 colunas num tempo de, aproximadamente, 206.86 segundos. No relatório é possível ver que, na solução, foi utilizado o método dual simplex e foram feitas 13828 iterações simplex para resolução. Por fim, ele conseguiu resolver o problema em 249.28 segundos encontrando uma solução ótima com um gap de 0.0000%.

VI. CONCLUSÃO

A praticidade de solução de um modelo com grandes instâncias e, sobretudo a iteração do *solver* com uma ferramenta como o python fez toda a diferença nessa pesquisa. Por muitas vezes o resultado pode ser demasiadamente abstrato, mas com as ferramentas certas e, mais do que isso, com a integração correta entre as plataformas, contribui com que o modelo possa ser facilmente modelado e facilmente interpretado para que possa ser implementado.

O nível de imersão que os estudos no campo de Pesquisa Operacional proporciona é algo que nos motivou a não nos limitarmos à uma simples construção de um modelo, mas a buscar melhorias que pudessem contornar empecilhos nas restrições, por exemplo. Obviamente existem muitas melhorias a serem feitas nesta modelagem, mas para os fins a que se propõe essa pesquisa saímos satisfeitos com os resultados encontrados e com a experiência de termos usado ferramentas importantes do campo de Pesquisa Operacional.

VII. REFERÊNCIAS

[1] Flamand, T., Iori, M., Haouari, M. (2023). The transportation problem with packing constraints. Computers & Operations Research, 157, 106278.

[2] Jalal, A., Toso, E. A. V., Morabito, R. (2023). A location–transportation problem under demand uncertainty for a pharmaceutical network in Brazil. Computers & Chemical Engineering, 174, 108233. K. Elissa, "Title of paper," unpublished.

[3] Arenales, M., Armentano, V. (2006). Pesquisa Operacional. Capa comum. Português.

[4] OpenAI. (n.d.). Recuperado de https://chat.openai.com/

[5] ScienceDirect. (s.d.). Recuperado de https://www.sciencedirect.com/