UNIVERSIDADE ABERTA
UNIVERSIDADE DE TRÁS-OS-MONTES E ALTO DOURO

Planeamento de Entregas

Hugo Gonçalves - 2100562

Pedro Morais - 2401849

João Valadares - 2401840

Mestrado em Engenharia Informática e Tecnologia Web

2024

Índice

1 Introdução **1**

2 Problema **1**

2.1 Cenário 2

2.2 Objetivo 2

2.3 Desafio 3

3 Formulação **3**

3.1 Variáveis 3

3.2 Função Objetivo 4

3.3 Restrições 5

3.4 Formulação Final 6

4 Implementação Computacional **7**

4.1 Bibliotecas Utilizadas 7

4.2 Estrutura do Código 8

4.3 Cenários Simulados 9

4.4 Execução e Resultados 9

5 Demonstração da Resolução e Resultados Obtidos **9**

5.1 Resultados da Resolução 9

5.2 Análise dos Resultados 10

5.3 Cenários Alternativos 11

5.4 Notas finais 11

Lista de Tabelas

5.1 Resultados da Resolução 10

1 Introdução

O presente relatório visa apresentar o desenvolvimento e a resolução de um problema prático de programação linear, uma área central na investigação operacional. Este trabalho foi realizado por um grupo composto por três estudantes, cada um assumindo funções específicas para garantir uma abordagem colaborativa e eficiente. Desde o início, foram estabelecidas as seguintes responsabilidades: João Valadares como líder, Pedro Morais como *designer*, e Hugo Gonçalves como analista. Esta organização permitiu estruturar o trabalho de forma eficaz, desde a definição do problema até à obtenção dos resultados.

A temática escolhida, o planeamento de entregas de uma empresa de logística, foi selecionada devido à sua relevância prática e impacto no dia a dia. O problema consiste em maximizar o número de encomendas entregues, utilizando uma frota de veículos com capacidades e velocidades variadas, bem como um número limitado de motoristas e restrições de tempo. Este cenário reflete os desafios reais enfrentados por empresas logísticas, como a necessidade de otimizar recursos para atender à demanda de forma eficiente.

Para a realização deste trabalho, foram utilizadas várias ferramentas tecnológicas, incluindo o Moodle para partilha de informações, o Microsoft Teams para reuniões e comunicação, o LaTeX para a elaboração do relatório, e o Google Colab para a implementação da programação e resolução do problema. Estas ferramentas facilitaram a colaboração em equipa e a análise sistemática dos dados.

Este relatório detalha todas as etapas do processo, desde a formulação do problema, definição das variáveis, função objetivo e restrições, até à apresentação dos resultados obtidos e conclusões. Pretende-se não apenas encontrar a solução ótima para o problema proposto, mas também explorar cenários alternativos que possam oferecer *insights* adicionais sobre a gestão de recursos e o impacto das condições impostas.

2 Problema

Uma empresa de logística enfrenta o desafio de planear a entrega de encomendas para otimizar o uso da sua frota e dos seus motoristas. O objetivo é maximizar o número de encomendas entregues, respeitando as restrições de capacidade dos veículos, disponibilidade de motoris-

tas, tempos de trabalho e distâncias a percorrer. Este problema é representativo das operações diárias enfrentadas por muitas empresas no setor de transportes e logística.

2.1 Cenário

A empresa dispõe de uma frota de cinco veículos, cada um com características distintas de capacidade e velocidade média:

- **Veículo A**: Capacidade de 5 m³, velocidade média de 50 km/h.
- **Veículo B** e **C**: Cada um com capacidade de 3 m³ e velocidade média de 70 km/h.
- **Veículo D** e **E**: Cada um com capacidade de 1 m³ e velocidade média de 80 km/h.

No início do planeamento, o armazém possui um inventário de 1000 encomendas, distribuídas em três categorias com diferentes volumes unitários:

- **Tipo A**: 500 encomendas, volume de 0,017 m³ por caixa.
- **Tipo B**: 250 encomendas, volume de 0,053 m³ por caixa.
- **Tipo C**: 250 encomendas, volume de 0,026 m³ por caixa.

Além disso, a empresa tem à disposição oito motoristas, cada um com um turno diário de 8 horas. Os veículos podem ser utilizados continuamente por diferentes motoristas ao longo das 24 horas do dia, desde que respeitadas as restrições de tempo de trabalho. Cada entrega envolve um percurso de 15 km, e a capacidade máxima de cada veículo não pode ser excedida em momento algum.

2.2 Objetivo

O objetivo deste planeamento é determinar a alocação ideal de motoristas e encomendas aos veículos, para maximizar o número de encomendas entregues, respeitando as seguintes restrições:

1. **Capacidade dos veículos**: O volume total das encomendas transportadas por cada veículo numa viagem não pode exceder a sua capacidade.

2. **Tempo de entrega**: Cada veículo deve cumprir as entregas dentro do tempo disponível, considerando a distância e sua velocidade média.

3. **Tempo máximo de uso dos veículos**: Cada veículo pode operar por, no máximo, 24 horas.

4. **Turnos dos motoristas**: Cada motorista pode trabalhar no máximo 8 horas por dia.

5. **Número de motoristas disponíveis**: O total de motoristas alocados não pode ultrapassar o limite disponível de 8 motoristas.

6. **Limitação de encomendas**: O número total de encomendas entregues não pode ultrapassar o inventário disponível de cada tipo.

2.3 Desafio

A tarefa consiste em determinar:

1. Quantas encomendas de cada tipo devem ser alocadas a cada veículo.

2. Quantos motoristas devem ser alocados a cada veículo.

3. O tempo de operação de cada veículo.

Este problema será resolvido com a aplicação de métodos de programação linear, para maximizar o número total de encomendas entregues dentro das restrições impostas.

3 Formulação

3.1 Variáveis

As variáveis utilizadas no problema foram definidas para representar as quantidades necessárias para a formulação do modelo de programação linear. Estas variáveis são descritas a seguir:

- x_{ij}: Número de encomendas do tipo i alocadas ao veículo j, onde:
  - $i \in \{1, 2, 3\}$: Representa os tipos de encomenda:

    * $i = 1$: Encomendas do tipo A ($0.017\,\text{m}^3$).
    * $i = 2$: Encomendas do tipo B ($0.053\,\text{m}^3$).
    * $i = 3$: Encomendas do tipo C ($0.026\,\text{m}^3$).
  - $j \in \{A, B, C, D, E\}$: Representa os veículos disponíveis:
    * A: Veículo com capacidade $5\,\text{m}^3$, velocidade $50\,\text{km/h}$.
    * B, C: Veículos com capacidade $3\,\text{m}^3$, velocidade $70\,\text{km/h}$.
    * D, E: Veículos com capacidade $1\,\text{m}^3$, velocidade $80\,\text{km/h}$.

- y_j: Número de motoristas alocados ao veículo j, onde $j \in \{A, B, C, D, E\}$.
- t_j: Tempo total de uso do veículo j (em horas), onde $j \in \{A, B, C, D, E\}$.

Adicionalmente, foram utilizadas constantes para representar os limites físicos e operacionais do problema:

- v_i: Volume de uma encomenda do tipo i (em m^3).
- C_j: Capacidade do veículo j (em m^3).
- v_j: Velocidade média do veículo j (em km/h).
- E_i: Número total de encomendas disponíveis do tipo i.
- T_{max}: Tempo máximo de uso diário de cada veículo ($24\,\text{horas}$).
- D: Distância necessária para cada entrega ($15\,\text{km}$).
- H_{turno}: Duração máxima do turno de cada motorista ($8\,\text{horas}$).
- M_{max}: Número total de motoristas disponíveis (8).

3.2 Função Objetivo

A formulação matemática do problema foi realizada para maximizar o número total de encomendas entregues, respeitando as restrições operacionais e de capacidade impostas. A função objetivo é definida como:

$$\text{Maximizar } Z = \sum_{i=1}^{3} \sum_{j \in \{A,B,C,D,E\}} x_{ij}$$

Onde:

- x_{ij}: Número de encomendas do tipo i alocadas ao veículo j.
- $i \in \{1, 2, 3\}$: Tipos de encomendas.
- $j \in \{A, B, C, D, E\}$: Veículos disponíveis.

3.3 Restrições

O modelo está sujeito às seguintes restrições:

1. **Capacidade dos veículos**: O volume total das encomendas transportadas por cada veículo não pode exceder a sua capacidade:

$$\sum_{i=1}^{3} vol_i \cdot x_{ij} \leq C_j, \quad \forall j \in \{A, B, C, D, E\}$$

   Onde:

   - vol_i: Volume de uma encomenda do tipo i (em m^3).
   - C_j: Capacidade do veículo j (em m^3).

2. **Tempo necessário para entrega**: O tempo total para as entregas de cada veículo não pode exceder o tempo disponível:

$$\sum_{i=1}^{3} \frac{x_{ij} \cdot D}{vel_j} \leq t_j, \quad \forall j \in \{A, B, C, D, E\}$$

   Onde:

   - D: Distância necessária para cada entrega ($15\,\text{km}$).
   - vel_j: Velocidade média do veículo j (km/h).
   - t_j: Tempo total de uso do veículo j (em horas).

3. **Tempo máximo por veículo**: O tempo de uso de cada veículo não pode exceder 24 horas:

$$t_j \leq 24, \quad \forall j \in \{A, B, C, D, E\}$$

4. **Turnos dos motoristas**: O tempo de uso de cada veículo deve estar limitado ao número de motoristas alocados multiplicado pelo turno máximo de trabalho:

$$t_j \leq y_j \cdot H_{\text{turno}}, \quad \forall j \in \{A, B, C, D, E\}$$

Onde:

- y_j: Número de motoristas alocados ao veículo j.
- H_{turno}: Duração máxima do turno de cada motorista (8 horas).

5. **Número total de motoristas**: O total de motoristas alocados não pode exceder o número disponível:

$$\sum_{j \in \{A,B,C,D,E\}} y_j \leq M_{\text{max}}$$

Onde $M_{\text{max}} = 8$ motoristas.

6. **Limitação de encomendas disponíveis**: O número total de encomendas entregues por tipo não pode exceder o inventário disponível:

$$\sum_{j \in \{A,B,C,D,E\}} x_{ij} \leq E_i, \quad \forall i \in \{1, 2, 3\}$$

Onde E_i: Total de encomendas disponíveis do tipo i.

7. **Restrições de não negatividade**: Todas as variáveis devem ser não negativas:

$$x_{ij} \geq 0, \quad y_j \geq 0, \quad t_j \geq 0, \quad \forall i, j$$

3.4 Formulação Final

Com base na função objetivo e nas restrições descritas anteriormente, a formulação final do problema de programação linear é apresentada a seguir:

$$\text{Maximizar } Z = \sum_{i=1}^{3} \sum_{j \in \{A,B,C,D,E\}} x_{ij}$$

s.a:

1.

$$\sum_{i=1}^{3} vol_i \cdot x_{ij} \leq C_j, \quad \forall j \in \{A, B, C, D, E\}$$

2.
$$\sum_{i=1}^{3} \frac{x_{ij} \cdot D}{vel_j} \leq t_j, \quad \forall j \in \{A, B, C, D, E\}$$

3.
$$t_j \leq 24, \quad \forall j \in \{A, B, C, D, E\}$$

4.
$$t_j \leq y_j \cdot H_{\text{turno}}, \quad \forall j \in \{A, B, C, D, E\}$$

5.
$$\sum_{j \in \{A,B,C,D,E\}} y_j \leq M_{\text{max}}$$

6.
$$\sum_{j \in \{A,B,C,D,E\}} x_{ij} \leq E_i, \quad \forall i \in \{1, 2, 3\}$$

$$x_{ij} \geq 0, \quad y_j \geq 0, \quad t_j \geq 0, \quad \forall i, j$$

4 Implementação Computacional

A resolução do problema foi realizada utilizando a linguagem de programação Python, em conjunto com a biblioteca OR-Tools, desenvolvida pela Google, que oferece suporte avançado para problemas de otimização, incluindo programação linear. O código desenvolvido foi projetado para resolver o problema de planeamento logístico descrito neste relatório.

4.1 Bibliotecas Utilizadas

A implementação utilizou as seguintes bibliotecas:

- **OR-Tools**: Biblioteca de código aberto amplamente utilizada para resolver problemas de otimização combinatória e programação linear. No problema apresentado, utilizou-se o módulo `pywraplp` para a resolução de problemas de programação linear inteira.
- **Python (Standard Library)**: Para cálculos matemáticos e estruturas de controlo básicas, como ciclos e funções.

4.2 Estrutura do Código

O código foi estruturado em diferentes etapas para garantir clareza e modularidade. Estas etapas são descritas a seguir:

1. **Inicialização do Solver:** Foi utilizado o solver `SCIP` (*Solving Constraint Integer Programs*), um dos solucionadores disponíveis na OR-Tools, ideal para problemas de programação linear inteira com múltiplas variáveis e restrições.

2. **Definição dos Dados do Problema:**

    - Capacidade dos veículos, velocidades médias e volumes das encomendas foram definidos como constantes.
    - Restrição de motoristas, tempos máximos de trabalho e disponibilidade dos veículos também foram especificados.

3. **Variáveis de Decisão:** Foram definidas as seguintes variáveis:

    - x_{ij}: Número de encomendas do tipo i alocadas ao veículo j.
    - y_j: Número de motoristas alocados ao veículo j.
    - t_j: Tempo total de uso do veículo j.
    - trips_j: Número de viagens realizadas por cada veículo.

4. **Função Objetivo:** A função objetivo foi definida para maximizar o número total de encomendas entregues, com uma prioridade maior para encomendas de menor volume, utilizando um fator de eficiência volumétrica.

5. **Definição das Restrições:** Foram implementadas todas as restrições descritas na formulação matemática, incluindo:

    - Limitações de capacidade volumétrica dos veículos.
    - Tempo necessário para completar as entregas.
    - Tempo máximo diário de uso dos veículos.
    - Disponibilidade e turnos dos motoristas.
    - Limitações no número de encomendas disponíveis.

6. **Resolução do Problema:** O solver foi executado para encontrar a solução ótima, com os resultados apresentados em termos do número de encomendas entregues, motoristas alocados e tempo de operação de cada veículo.

4.3 Cenários Simulados

O código também permite a simulação de cenários alternativos, com ajustes nos seguintes parâmetros:

- Número de motoristas disponíveis (`max_drivers`).
- Duração máxima dos turnos de trabalho (`driver_shift_time`).
- Inclusão de veículos adicionais com características específicas.

Estas simulações foram realizadas iterativamente para explorar o impacto de mudanças nos recursos sobre o número total de encomendas entregues.

4.4 Execução e Resultados

O código desenvolvido foi testado e executado para diferentes cenários, proporcionando uma análise detalhada das soluções possíveis e identificando as principais limitações no modelo inicial. O uso da OR-Tools garantiu soluções eficientes, mesmo para cenários mais complexos com múltiplas restrições e variáveis.

5 Demonstração da Resolução e Resultados Obtidos

5.1 Resultados da Resolução

A solução ótima encontrada para as condições iniciais é apresentada na Tabela 5.1.

Veículo	Utilização	Tipo de Encomenda	Encomendas Entregues	Motoristas Alocados
A	0h	Nenhum	0	0
B	0h	Nenhum	0	0
C	16h	Tipo 1	74	2
D	24h	Tipo 1	128	3
E	24h	Tipo 1	128	3

Tabela 5.1: Resultados da Resolução

5.2 Análise dos Resultados

Os resultados mostram que, com os recursos disponíveis (5 veículos e 8 motoristas), foi possível entregar um total de 330 encomendas, todas do tipo 1 (menor volume). Este cenário evidencia algumas características importantes:

- **Utilização dos veículos:** Apenas os veículos C, D e E foram utilizados. O veículo A, apesar de ter a maior capacidade ($5\,\mathrm{m}^3$), não foi usado devido à sua velocidade reduzida ($50\,\mathrm{km/h}$), tornando-o ineficiente para maximizar entregas no tempo disponível. O veículo B, com características similares ao veículo C, não foi utilizado por falta de motoristas disponíveis.

- **Alocação dos motoristas:** A totalidade dos motoristas foi alocada para os veículos C, D e E, priorizando os veículos menores e mais rápidos ($70 - 80\,\mathrm{km/h}$). Isso permitiu maximizar o número de viagens e, consequentemente, as entregas.

- **Tipos de encomendas entregues:** Apenas encomendas do tipo 1 ($0.017\,\mathrm{m}^3$) foram entregues, pois possuem o menor volume, permitindo transportar mais unidades por viagem. Encomendas dos tipos 2 e 3 não foram entregues, indicando que, com as restrições atuais, não é viável priorizá-las.

- **Eficiência por veículo:**

  - O veículo C realizou 16 horas de operação e entregou 74 encomendas.
  - Os veículos D e E operaram por 24 horas cada e entregaram 128 encomendas cada.

5.3 Cenários Alternativos

Para explorar a viabilidade de aumentar o número total de encomendas entregues, foram simulados cenários alternativos com ajustes nos recursos disponíveis:

- **Aumento de motoristas:** Adicionando mais 2 motoristas (totalizando 10) e permitindo turnos de 12 horas, o número de encomendas entregues aumentou para 560. Esse cenário evidencia que a limitação de motoristas foi um fator crítico no modelo inicial.
- **Aumento da frota:** Com a inclusão de mais 4 veículos semelhantes aos D e E, e elevando o total de motoristas para 20, foi possível entregar todas as 1000 encomendas disponíveis. Este cenário demonstra a necessidade de ampliar significativamente os recursos para atender à totalidade da demanda.

5.4 Notas finais

Os resultados obtidos destacam os seguintes pontos:

- O número de motoristas é a principal limitação no cenário inicial, impedindo o uso pleno de todos os veículos disponíveis.
- O foco em encomendas menores (tipo 1) foi estratégico para maximizar a eficiência volumétrica, mas limitou a diversidade de encomendas entregues.
- Cenários com mais motoristas e veículos menores demonstraram ser mais eficazes para atender a demandas elevadas, com base na priorização do uso de recursos rápidos e flexíveis.

Anexo A

O relatório original em LaTeX está disponível neste repositório.

Anexo B

Vídeo de apresentação do trabalho disponível aqui.

Anexo C

Abaixo apresenta-se o código completo utilizado para a resolução do problema de planejamento logístico. O código foi ajustado para suportar diferentes cenários e incluir o uso de recursos adicionais.

```
from ortools.linear_solver import pywraplp

def solve_logistics_with_ortools(
    max_drivers=8,
    driver_shift_time=8,
    capacities={'A': 5, 'B': 3, 'C': 3, 'D': 1, 'E': 1},
    speeds={'A': 50, 'B': 70, 'C': 70, 'D': 80, 'E': 80}
):
  # Criar o solver
  solver = pywraplp.Solver.CreateSolver('SCIP') # Solving Constraint
      Integer Programs

  if not solver:
    print("Solver não disponível.")
    return

  # Dados do problema
  volumes = {1: 0.017, 2: 0.053, 3: 0.026}

  max_vehicle_time = 24  # Cada veículo pode operar por até 24 horas
  distance_per_leg = 15  # Apenas ida (não há volta)

```

```
22  # Capacidade máxima de encomendas por viagem e viagens possíveis
23  max_trips = {j: (max_vehicle_time * speeds[j]) // distance_per_leg for
        j in
24                capacities.keys()}
25  max_orders_per_trip = {
26    j: {i: int(capacities[j] / volumes[i]) for i in volumes.keys()} for
          j in
27    capacities.keys()}
28  max_orders = {1: 500, 2: 250, 3: 250}
29
30  # Variáveis de decisão
31  x = {}  # Número de encomendas do tipo i alocadas ao veículo j
32  for i in range(1, 4):
33    for j in capacities.keys():
34      x[i, j] = solver.IntVar(0, solver.infinity(), f'x[{i},{j}]')
35
36  trips = {j: solver.IntVar(0, max_trips[j], f'trips[{j}]') for j in
37           capacities.keys()}  # Número de viagens
38  y = {j: solver.IntVar(0, max_drivers, f'y[{j}]') for j in
39       capacities.keys()}  # Motoristas
40  vehicle_time = {j: solver.NumVar(0, max_vehicle_time,
        f'vehicle_time[{j}]')
41                  for j in capacities.keys()}  # Horas totais de uso do
                        veículo
42
43  # Função objetivo: Maximizar o número total de encomendas entregues
        com prioridade para encomendas menores
44  objective = solver.Objective()
45  for i in range(1, 4):
46    for j in capacities.keys():
47      priority = 1 / volumes[
48        i]  # Priorização com base na eficiência volumétrica
49      objective.SetCoefficient(x[i, j], priority)
50  objective.SetMaximization()
51
52  # Restrições
53  # 1. Respeitar a capacidade máxima por viagem e número total de
```

```
    encomendas entregues
for j in capacities.keys():
  # Capacidade volumétrica por viagem
  for i in range(1, 4):
    solver.Add(x[i, j] <= trips[j] * max_orders_per_trip[j][i])

  # Total de encomendas não pode exceder o limite físico de encomendas
  solver.Add(solver.Sum(x[i, j] for i in volumes.keys()) <= trips[j] *
      sum(
       max_orders_per_trip[j].values()))

# 2. Respeitar o tempo necessário para entregar as encomendas
for j in capacities.keys():
  solver.Add(vehicle_time[j] >= solver.Sum(
      x[i, j] * (distance_per_leg / speeds[j]) for i in
          volumes.keys()))

# 3. Respeitar o tempo máximo de uso por veículo em 24 horas
for j in capacities.keys():
  solver.Add(vehicle_time[j] <= max_vehicle_time)

# 4. Tempo de trabalho dos motoristas por turno
for j in capacities.keys():
  solver.Add(vehicle_time[j] <= y[j] * driver_shift_time)

# 5. Respeitar o número máximo de encomendas disponíveis
for i in range(1, 4):
  solver.Add(
    solver.Sum(x[i, j] for j in capacities.keys()) <= max_orders[i])

# 6. Limitar o número total de motoristas disponíveis
solver.Add(solver.Sum(y[j] for j in capacities.keys()) <= max_drivers)

# Resolver o problema
status = solver.Solve()

# Resultados
```

```
88    if status == pywraplp.Solver.OPTIMAL:
89      print("Solução ótima encontrada!")
90      total_orders = 0
91      orders_by_driver = {j: 0 for j in capacities.keys()}
92
93      for i in range(1, 4):
94        for j in capacities.keys():
95          total_orders += x[i, j].solution_value()
96          orders_by_driver[j] += x[i, j].solution_value()
97          print(
98              f"Número de encomendas do tipo {i} alocadas ao veículo {j}:
                    {x[i, j].solution_value()}")
99      print()
100     for j in capacities.keys():
101       print(
102           f"Número de motoristas alocados ao veículo {j}:
                  {y[j].solution_value():.0f}")
103       total_time = trips[j].solution_value() * (distance_per_leg /
              speeds[j])
104       print(
105           f"Tempo total de disponibilidade do veículo {j}:
                  {total_time:.0f} horas")
106       print(
107           f"Tempo total de uso do veículo {j}:
                  {vehicle_time[j].solution_value():.0f} horas")
108       if y[j].solution_value() > 0:
109         print(
110             f"Número de encomendas entregues pelo veículo {j}:
                    {orders_by_driver[j]:.0f}")
111       print()
112     print(f"Total de encomendas entregues: {total_orders}")
113   else:
114     print("Não foi possível encontrar uma solução ótima.")
115
116
117 # Executar o solver com OR-Tools
118 solve_logistics_with_ortools()
```

```
119
120 capacities = {'A': 5, 'B': 3, 'C': 3, 'D': 1, 'E': 1}
121 speeds = {'A': 50, 'B': 70, 'C': 70, 'D': 80, 'E': 80}
122
123 for drivers in range(8, 21):
124   for shift_time in range(8, 13):
125     solve_logistics_with_ortools(drivers, shift_time, capacities, speeds)
126
127
128 capacities = {'A': 5, 'B': 3, 'C': 3, 'D': 1, 'E': 1, 'F': 1, 'G': 1,
      'H': 1, 'I': 1}
129 speeds = {'A': 50, 'B': 70, 'C': 70, 'D': 80, 'E': 80, 'F': 80, 'G': 80,
      'H': 80, 'I': 80}
130
131 for drivers in range(20, 21):
132   for shift_time in range(8, 13):
133     solve_logistics_with_ortools(drivers, shift_time, capacities, speeds)
```