tp01-horario

September 29, 2023

1 Trabalho 1 - Horário

2 de outubro de 2023

Paulo Jorge Fernandes Freitas - A100053 & Pedro Manuel Pereira dos Santos - A100110

1.1 Enunciado:

Pretende-se construir o horário semanal de aulas de uma turma.
1. Existe um conjunto de salas S classificadas em "grandes" e "pequenas".
2. O tempo do horário está organizado em "slots" de uma hora. O total do tempo disponível é 5 horas de manhã e 5 horas às tarde.
3. Existe um conjunto D de disciplinas. Cada disciplina tem um atributo d com valor 1 ou 2, que classifica a duração de cada sessão (um ou dois "slots") , um atributo a entre 2 e 3 que define o número de sessões semanais e um atributo s entre 0 e 1 que diz se a sessão necessita de uma sala grande ou não.
4. Existe um conjunto P de professores. Cada professor tem associado um conjunto h das disciplinas que está habilitado a lecionar.
5. O horário está organizado em sessões concorrentes onde cada sessão é definido por uma disciplina desce que salas e professores verifiquem as seguintes restrições:
1. Para cada disciplina todas as aulas decorrem na mesma sala e com o mesmo professor.
2. O número total de horas lecionadas por cada professor está num intervalo de mais ou menos 20% do número médio de horas lecionadas pela totalidade dos professores.
3. Nenhuma sala pode ser ocupada simultaneamente por mais do que uma aula e nenhum professor pode lecionar simultaneamente mais do que uma aula.
4. Em cada disciplina, cada aula é lecionada por um professor habilitado para essa disciplina e ocorre numa sala de tamanho apropriado à disciplina.

Use o package ortools para encontrar uma solução que verifique todas as restrições e maximize o número de partes de dia (manhãs ou tardes) que estão livres de qualquer aula.

1.2 Análise do Problema

Este é um problema de alocação. Queremos alocar aulas de diferentes disciplinas a um limitado numero de slots durante uma semana. Essa alocação implica a alocação de professores e salas aprorpiados a cada disciplina.

Existem salas de aulas grandes e pequenas, descritas separadamente em listas com os seus nomes.

Existe uma lista com os dias da semana, para facilitar o "printer" e facilitar a leitura do output.

Há duas variáveis para slots : "slots_manha" que indica o número de slots existentes na primeira metade do dia, e "slots_tarde" que corresponde aos restantes slots do dia.

A variável "professores" descreve os professores existentes numa lista. Para seu auxílio, a variável "leciona" estabelece, recorrendo a um dicionário, uma lista de quais disciplinas os docentes estam habilitados a lecionar.

Na variável "disciplinas", são enumeradas numa lista as disciplinas existentes. Em seu auxílio, o dicionário "aulas" estabelece quais os atributos as aulas requerem. Num tuplo são indicados: o número de slots que a aula ocupa, o número de aulas por semana e o se é necessarío uma sala grande ou sala pequena, respetivamente.

Existe tambem uma variável binária "horario" capaz de relacionar disciplinas,professores, salas, dias e slots onde:

$x_{d,p,s,dia,slot} == 1$ se e só se é possivel alocar aula d com o professor p numa sala s, no dia dia e no momento

Variáveis:

salas_g - Salas grande / Sg
salas_p - Salas pequenas / Sp
(salas_p + salas_g) - Salas / S
slot - hora / H
disciplina - Disciplinas / D
professores - Professores / P
dias - Dias / X

1.3 Iniciação

Para a resolução deste exercício utilizamos a biblioteca OR-Tools que criou uma interface para o SCIP. Esta biblioteca foi instalada com o commando pip install ortools.

[3]:
```
!pip install ortools
```

```
Requirement already satisfied: ortools in /usr/local/lib/python3.10/dist-
packages (9.7.2996)
Requirement already satisfied: absl-py>=0.13 in /usr/local/lib/python3.10/dist-
packages (from ortools) (1.4.0)
Requirement already satisfied: numpy>=1.13.3 in /usr/local/lib/python3.10/dist-
packages (from ortools) (1.23.5)
Requirement already satisfied: protobuf>=4.23.3 in
/usr/local/lib/python3.10/dist-packages (from ortools) (4.24.3)
```

1.4 Implementação

Começamos por importar a biblioteca de programação linear do OR-Tools e criar uma instância do solver.

Depois inicializamos o solver, definimos as variaveis.

[4]:
```
from ortools.linear_solver import pywraplp
```

```
solver = pywraplp.Solver.CreateSolver('SCIP')

salas_g = ["a1","a2"] # auditorios - salas grandes
salas_p = ["s1","s2","s3","s4"] # salas pequenas
dias = ["Segunda","Terça","Quarta","Quinta","Sexta"]
slots_manha = 5
slots_tarde = 5
professores = ["p1","p2","p3","p4","p5"]
disciplinas = ["d1","d2","d3","d4","d5"]

aulas = {"d1":(2,2,1), "d2":(1,3,0), "d3":(2,3,1), "d4":(2,2,0), "d5":(1,3,1)}
 ↪# quais sao os atributos das cadeiras em formato (n de slots a ocupar, n de
 ↪sessões, se precisa de sala grande-1 ou não-0)

leciona = {"p1":["d1","d3"], "p2":["d3","d4"], "p3":["d2"], "p4":["d4"], "p5" :
 ↪["d5"]} # quais cadeiras os professores podem lecionar

horario={}
for d in disciplinas:
  for p in professores:
    for s in salas_g + salas_p:
      for dia in dias:
        for slot in range(slots_manha + slots_tarde):
          horario[d,p,s,dia,slot]= solver.
 ↪IntVar(0,1,f'x[{d},{p},{s},{dia},{slot}]')
```

1.4.1 Restrições

Vamos agora adicionar as restrições. Para tal, vamos dividir as condições no enunciado para facilitar a criação da expressão lógica e a interpretação.

1. Cada disciplina tem um dado número de aulas semanais. ($ A_d = aulas[d][0]* aulas[d][1])$

$$\forall_{d \leq D} \sum_{p \leq P, s \leq S x \leq X, h \leq H} horario_{d,p,s,x,h} = A_d$$

Isto se dá pois algumas disciplinas têm aulas que ocupam dois slots cada, e deste modo temos todos os slots semanais que a disciplina ocupa.

[5]:
```
#1 Cada disciplina tem um dado numero de aulas semanais
for d in disciplinas:
    naulas = aulas[d][1]
    nslots = aulas[d][0]
    solver.Add(solver.Sum(horario[d, p, s, dia, slot] for p in professores for
 ↪s in salas_g + salas_p for dia in dias for slot in range(slots_manha +
 ↪slots_tarde)) == nslots*naulas)
```

2. Professores não podem dar mais de uma aula no mesmo slot e nenhuma sala pode ser ocupada por mais de uma aula no mesmo slot.

$$\forall_{p \leq P} \cdot \forall_{s \leq S} \sum_{d \leq D, x \leq X, h \leq H} horario_{d,p,s,x,h} <= 1$$

Impede que atribuia aulas no mesmo slot para o mesmo professor e, juntamente impede que ocorram aulas, no mesmo slot, na mesma sala.

[6]:
```
#2 Professores não podem dar mais de uma aula no mesmo slot e nenhuma sala pode␣
↪ser ocupada por mais de uma aula no mesmo slot
for dia in dias:
    for slot in range(slots_manha + slots_tarde):
        for s in salas_g + salas_p:
            solver.Add(solver.Sum(horario[d, p, s, dia, slot] for d in␣
↪disciplinas for p in professores) <= 1)
```

3. Cada aula deve ser lecionada em uma sala adequada. ($ B_d = aulas[d][2])$

$$\forall_{d \leq D} \cdot \forall_{s \leq S} \sum_{p \leq P, x \leq X, h \leq H} horario_{d,p,s,x,h} <= B_d$$

Esta restrição verifica qual o tamanho de sala requerido, e impede que as aulas sejam lecionadas em salas de tamanho errado.

[7]:
```
#3 Cada aula deve ser dada em uma sala adequada
for d in disciplinas:
    for dia in dias:
        for slot in range(slots_manha + slots_tarde):
            # Se a disciplina requer uma sala grande (1), então ela só pode ser␣
↪agendada em salas grandes
            if aulas[d][2] == 1:
                solver.Add(solver.Sum(horario[d, p, s, dia, slot] for p in␣
↪professores for s in salas_p) == 0)
            # Se a disciplina não requer uma sala grande (0), então ela só pode␣
↪ser agendada em salas pequenas
            else:
                solver.Add(solver.Sum(horario[d, p, s, dia, slot] for p in␣
↪professores for s in salas_g) == 0)
```

4. As aulas devem ser lecionadas por professores habilitados. ($ C_p = leciona[p])$

$$\forall_{d<D} . \forall_{p<P} . \forall_{s<S} . \forall_{x<X} . \forall_{h<H} d \notin C_p \implies horario_{d,p,s,x,h} = 0$$

Esta restrição verifica quais as disciplinas o docente pode lecionar, e impede que professores lecionem disciplinas que não podem.

[8]:
```
#4 As aulas devem ser lecionadas por professores habilitados
for p in professores:
    for d in disciplinas:
      if d not in leciona[p]:
        solver.Add(solver.Sum(horario[d, p, s, dia, slot] for dia in dias for s␣
 ↪in salas_p+salas_g for slot in range(slots_manha+slots_tarde))==0)
```

5. Apenas uma aula é dada em cada slot.

$$\forall_{x \leq X} \cdot \forall_{h \leq H} \sum_{d \leq D, p \leq P, s \leq S} horario_{d,p,s,x,h} <= 1$$

Impede que mais do que uma aula ocorre no mesmo slot de tempo.

[9]:
```
#5 Apenas uma aula é dada em cada slot
for dia in dias:
    for slot in range(slots_manha + slots_tarde):
        solver.Add(solver.Sum(horario[d, p, s, dia, slot] for d in disciplinas␣
 ↪for p in professores for s in salas_g + salas_p) <= 1)
```

1.4.2 Procura da solução do problema

[10]:
```
problema = solver.Sum(1 - solver.Sum(horario[d, p, s, dia, slot] for d in␣
 ↪disciplinas for p in professores for s in salas_g + salas_p for dia in dias␣
 ↪for slot in range(slots_manha + slots_tarde)) for dia in dias for slot in␣
 ↪range(slots_manha + slots_tarde))
solver.Maximize(problema)

# Resolva o problema
status = solver.Solve()
```

1.4.3 Print da solução do problema

[11]:
```
# Verificar o status da solução
if status == pywraplp.Solver.OPTIMAL:
    print('Solução ótima encontrada:')
    for dia in dias:

      print("\n",dia)
      for slot in range(slots_manha + slots_tarde):
        if slot ==0: print("Manha")
        if slot ==5: print("Tarde")
        print("slot:",slot+1)
        for p in professores:
            for d in disciplinas:
                for s in salas_g + salas_p:
                    if horario[d, p, s, dia, slot].solution_value():
```

```
                        print(f'Disciplina {d}, Professor {p}, Sala {s}')
        print()
else:
    print('O solver não conseguiu encontrar uma solução ótima. Status:')
```

```
Solução ótima encontrada:

 Segunda
Manha
slot: 1
Disciplina d5, Professor p5, Sala a2

slot: 2
Disciplina d1, Professor p1, Sala a1

slot: 3

slot: 4

slot: 5

Tarde
slot: 6

slot: 7
Disciplina d4, Professor p2, Sala s3

slot: 8
Disciplina d3, Professor p2, Sala a2

slot: 9

slot: 10
Disciplina d3, Professor p1, Sala a2


 Terça
Manha
slot: 1

slot: 2
Disciplina d5, Professor p5, Sala a1

slot: 3

slot: 4
Disciplina d3, Professor p1, Sala a2
```

```
slot: 5
Disciplina d3, Professor p1, Sala a2

Tarde
slot: 6

slot: 7
Disciplina d5, Professor p5, Sala a2

slot: 8

slot: 9

slot: 10
Disciplina d2, Professor p3, Sala s2


 Quarta
Manha
slot: 1
Disciplina d2, Professor p3, Sala s1

slot: 2

slot: 3

slot: 4

slot: 5

Tarde
slot: 6

slot: 7
Disciplina d2, Professor p3, Sala s1

slot: 8

slot: 9
Disciplina d3, Professor p2, Sala a2

slot: 10
Disciplina d1, Professor p1, Sala a1


 Quinta
Manha
```

```
slot: 1

slot: 2
Disciplina d3, Professor p1, Sala a1

slot: 3

slot: 4
Disciplina d1, Professor p1, Sala a1

slot: 5

Tarde
slot: 6

slot: 7
Disciplina d4, Professor p2, Sala s4

slot: 8

slot: 9

slot: 10
Disciplina d1, Professor p1, Sala a1


 Sexta
Manha
slot: 1
Disciplina d4, Professor p2, Sala s3

slot: 2

slot: 3
Disciplina d4, Professor p4, Sala s4

slot: 4

slot: 5

Tarde
slot: 6

slot: 7

slot: 8

slot: 9
```

```
slot: 10
```