tp1-problema-1

October 8, 2024

1 TP1 - Problema 1

Grupo 23 Pedro Gonçalves a101250 José Loureiro a96467 Bruno Neiva a95311

Problema 1: Pretende-se construir um horário semanal para o plano de reuniões de projeto de uma "StartUp" de acordo com as seguintes condições: a. Cada reunião ocupa uma sala (enumeradas 1…S) durante um "slot" (tempo,dia). Assume-se os dias enumerados 1..D e, em cada dia, os tempos enumerados 1..T. b. Cada reunião tem associado um projeto (enumerados 1…P) e um conjunto de participantes. Os diferentes colaboradores são enumerados de 1…C. c. Cada projeto tem associado um conjunto de colaboradores, dos quais um é o líder. Cada projeto realiza um dado número de reuniões semanais. d. O líder do projeto participa em todas as reuniões do seu projeto; os restantes colaboradores podem ou não participar consoante a sua disponibilidade, num mínimo ("quorum") de 50% do total de colaboradores do projeto. A disponibilidade de cada participante, incluindo o lider, é um conjunto de "slots" ("inputs" do problema).

"Inputs" do problema: 1. Os parâmetros S, D, T, P e C. 2. O número mínimo de reuniões semanais, o conjunto de colaboradores de cada projeto, assim como o seu lider. 3. A disponibilidade de cada colaborador, incluindo o lider. Esta disponibilidade será um conjunto de "slots" representada por uma matriz booleana.

Optimização: 1. Maximizar o número de reuniões realizadas.

[48]:
```
from ortools.linear_solver import pywraplp
horario = pywraplp.Solver.CreateSolver("SCIP")
#Exemplo 1 (caso impossível)
'''S, D, T, P, C = 3, 5, 8, 1, 3
#3 salas, 5 dias, 8 horas, 1 projeto, 3 colaboradores
colab_por_proj = {0 : [0,1,2]}
lider_proj = {0 : 1}
n_reunioes = {0 : 1}

disponibilidade = {0 : [(0,0)],
                   1 : [(1,3), (2,2)],
                   2 : [(1,3), (2,4)]
                   }'''
#Exemplo 2
S, D, T, P, C = 2, 5, 7, 4, 4
colab_por_proj = {
    0: [0,1,3],
```

```
    1: [0,1,2],
    2: [1,2,3],
    3: [0,2,3]
}
lider_proj = {0: 0, 1: 1, 2: 2, 3: 3}
n_reunioes = {0: 10, 1: 6, 2: 12, 3: 10}
disponibilidade = {
0:[(1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 7), (2, 1), (2, 2), (2, 3), (2,␣
 ↪4), (2, 5), (2, 6), (2, 7), (3, 1),
(3, 2), (3, 3), (3, 4), (3, 5),(3, 6), (4, 1), (4, 2), (4, 3), (4, 4), (4, 5),␣
 ↪(4, 6), (4, 7)],
1:[(1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (1, 7), (2, 1), (2, 2), (2,␣
 ↪3), (2, 4), (2, 5), (2, 6), (2, 7),
(3, 1), (3, 2), (3, 3), (3, 5), (3, 6), (3, 7), (4, 1), (4, 3), (4, 4), (4, 5),␣
 ↪(4, 6), (4, 7)],
2:[(1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (1, 7), (2, 1), (2, 3), (2,␣
 ↪4), (2, 5), (2, 6), (2, 7), (3, 1),
(3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (3, 7), (4, 1), (4, 2), (4, 3), (4, 4),␣
 ↪(4, 5), (4, 6), (4, 7)],
3:[(1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (1, 7), (2, 1), (2, 2), (2,␣
 ↪3), (2, 4), (2, 5), (2, 6), (2, 7),
(3, 1), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (3, 7), (4, 1), (4, 2), (4, 3),␣
 ↪(4, 4), (4, 5), (4, 6),
(4, 7)]
}

#Inicializar as matrizes
x = {}
for s in range(S):
    x[s] = {}
    for d in range(D):
        x[s][d] = {}
        for h in range(T):
            x[s][d][h] = {}
            for p in range(P):
                x[s][d][h][p] = horario.BoolVar(f"x[{s}][{d}][{h}][{p}]")

y = {}
for c in range(C):
    y[c] = {}
    for d in range(D):
        y[c][d] = {}
        for h in range(T):
            y[c][d][h] = {}
            for p in range(P):
                y[c][d][h][p] = horario.BoolVar(f"y[{c}][{d}][{h}][{p}]")
```

Iremos agora modelar e introduzir as restrições no solver:

1. Cada reunião tem de ter 50% ou mais participação dos seus colaboradores.

$$\forall_{d<D}\ \forall_{p<P}\ \forall_{h<T}\ \sum_{c<C} y_{c,d,h,p} \geq 0.5 \cdot len(colabPorProj[p])$$

[49]:
```
for d in range(D):
    for p in range(P):
        for t in range(T):
            lider = lider_proj[p]
            colabs = colab_por_proj[p]
            horario.Add(sum([y[c][d][t][p] for c in colabs]) >= 0.
 ↪5*len(colabs)*y[lider][d][t][p])
```

2. Certificar que o líder participa em todas as reuniões do seu projeto.

$$\forall_{d<D}\ \forall_{p<P}\ \forall_{h<T} \sum_{s<S} x_{s,d,h,p} == y_{lider,d,h,p}$$

[50]:
```
for d in range(D):
    for p in range(P):
        for t in range(T):
            lider = lider_proj[p]
            horario.Add(sum([x[s][d][t][p] for s in range(S)]) ==␣
 ↪y[lider][d][t][p])
```

3. Os colaboradores apenas podem participar nas reuniões quando têm disponibilidade.

$$\forall_{p<P}.\forall_{c<C}.\forall_{d<D}.\forall_{h<T} \cdot (h,d) \notin disponibilidade(c) \implies y_{c,d,h,p} == 0$$

[51]:
```
for d in range(D):
    for t in range(T):
        for p in range(P):
            for c in range(C):
                if (d,t) not in disponibilidade[c]:
                    horario.Add(y[c][d][t][p] == 0)
```

4. Verificar se é satisfeito o número de reuniões semanais.

$$\forall_{p<P} \sum_{s<S,d<D,h<T} x_{s,d,h,p} \geq nReuioes[p]$$

[52]:
```
for p in range(P):
    reunioes = n_reunioes[p]
```

```
    horario.Add(sum([x[s][d][t][p] for s in range(S) for d in range(D) for t in␣
↪range(T)]) == reunioes)
```

5. Verificar que cada colaborador não está em mais do que uma reunião ao mesmo tempo.

$$\forall_{c<C}\ \forall_{d<D}\ \forall_{h<T} \sum_{p<P} y_{c,d,h,p} \leq 1$$

[53]:
```
for c in range(C):
    for d in range(D):
        for t in range(T):
            horario.Add(sum([y[c][d][t][p] for p in range(P)]) <= 1)
```

6. Cada sala apenas só pode ser usada para uma reunião de cada vez.

$$\forall_{s<S}\ \forall_{d<D}\ \forall_{h<T} \sum_{p<P} x_{s,d,h,p} \leq 1$$

[54]:
```
for s in range(S):
    for d in range(D):
        for t in range(T):
            horario.Add(sum([x[s][d][t][p] for p in range(P)]) <= 1)
```

7. Certificar que apenas os colaboradores de um projeto participam na sua reunião.

$$\forall_{c<C}\ \forall_{d<D}\ \forall_{h<T}\ \forall_{p<P} \cdot c \notin colabPorProj_p \implies yc, d, h, p == 0$$

[55]:
```
for c in range(C):
    for d in range(D):
        for t in range(T):
            for p in range(P):
                if c not in colab_por_proj[p]:
                    horario.Add(y[c][d][t][p] == 0)
```

Execução do Solver e formulação da tabela com o horário das reuniões.

[56]:
```
from tabulate import tabulate

status = horario.Solve()
print(status)
if status== pywraplp.Solver.OPTIMAL:
    head = ["Dia %i" % d for d in range(D)]
    head.insert(0, "Slots")
    h = [[] for x in range(T)]
    for t in range(T):
        h[t].insert(0, "Slot %i" % t)
```

```
    for d in range(D):
        for t in range(T):
            h[t].insert(d, "")
            for p in range(P):
                for s in range(S):
                    if round(x[s][d][t][p].solution_value()) == 1:
                        h[t][d] += ("*Projeto %i - sala %i\n Colab: " % (p,s))
                        for c in range(C):
                            if round(y[c][d][t][p].solution_value()) == 1:
                                h[t][d] += ("%i, " % c)
                        h[t][d] = h[t][d][:-2]
                        h[t][d] += ("\n\n")
    print(tabulate(h, headers=head))

else:
    print("Não foi encontrada solução")
```

```
0
Slots    Dia 0                Dia 1                Dia 2                Dia 3
Dia 4
-------  -------------------  -------------------  -------------------
-------------------  -------
             Slot 0
         *Projeto 0 - sala 0  *Projeto 0 - sala 0  *Projeto 2 - sala 0  *Projeto
2 - sala 0  Slot 1
          Colab: 0, 3          Colab: 0, 3          Colab: 1, 2          Colab:
1, 2

         *Projeto 1 - sala 1  *Projeto 1 - sala 1  *Projeto 3 - sala 1  *Projeto
3 - sala 1
          Colab: 1, 2          Colab: 1, 2          Colab: 0, 3          Colab:
0, 3
         *Projeto 0 - sala 0  *Projeto 0 - sala 0  *Projeto 2 - sala 0  *Projeto
2 - sala 0  Slot 2
          Colab: 0, 3          Colab: 0, 1, 3       Colab: 1, 2          Colab:
2, 3

         *Projeto 1 - sala 1                       *Projeto 3 - sala 1
          Colab: 1, 2                               Colab: 0, 3
         *Projeto 0 - sala 0  *Projeto 0 - sala 0  *Projeto 2 - sala 0  *Projeto
2 - sala 0  Slot 3
          Colab: 0, 3          Colab: 0, 3          Colab: 1, 2          Colab:
1, 2

         *Projeto 1 - sala 1  *Projeto 2 - sala 1  *Projeto 3 - sala 1  *Projeto
3 - sala 1
          Colab: 1, 2          Colab: 1, 2          Colab: 0, 3          Colab:
```

```
0, 3
         *Projeto 0 - sala 0  *Projeto 0 - sala 0                        *Projeto
3 - sala 0  Slot 4
          Colab: 0, 3          Colab: 0, 3                                Colab:
2, 3

         *Projeto 1 - sala 1  *Projeto 2 - sala 1
          Colab: 1, 2          Colab: 1, 2
         *Projeto 0 - sala 0  *Projeto 0 - sala 0  *Projeto 2 - sala 0
Slot 5
          Colab: 0, 3          Colab: 0, 3          Colab: 1, 2

         *Projeto 1 - sala 1  *Projeto 2 - sala 1  *Projeto 3 - sala 1
          Colab: 1, 2          Colab: 1, 2          Colab: 0, 3
         *Projeto 3 - sala 0  *Projeto 2 - sala 0  *Projeto 2 - sala 0
Slot 6
          Colab: 2, 3          Colab: 1, 2          Colab: 1, 2

                              *Projeto 3 - sala 1  *Projeto 3 - sala 1
                               Colab: 0, 3          Colab: 0, 3
```