TP1

Propósito do trabalho

O propósito deste trabalho é a análise de problemas de alocação usando técnicas de SAT, em lógica proposicional, e IP em lógica linear inteira.

Enunciado

Problema 1

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" $1..T$ $(\text{hora}, \text{dia})$.
   B. Cada reunião tem associado um projeto (enumerados $1..P$) e um conjunto de participantes. Os diferentes colaboradores são enumerados $1..C$.
   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.

   São "inputs" do problema:

   A. Os parâmetros S, T, P, C
   B. O conjunto de colaboradores de cada projeto, o seu líder e o número de reuniões semanais.
   C. A disponibilidade de cada participante, incluindo o líder. Essa disponibilidade é um conjunto de "slots" representada numa matriz booleana de acessibilidade com uma linha por cada participante $1..C$ e uma coluna por "slot" $1..T$

   São critérios de optimização:

   A. Maximizar o número de reuniões efetivamente realizadas
   B. Minimizar o número médio de reuniões por participante.

Resolução

Variáveis do programa

```
S - Número de salas
P - Número de projetos
C - Número de colaboradores
T - Lista de slots (dia, hora)
CP - Dicionário em que as chaves são o id dum projeto e o valor correspondente é a lista de colaboradores associados ao projeto
LP - Dicionário em que as chaves são o id dum projeto e o valor correspondente é o id do líder desse projeto
projects_in - Dicionário em que as chaves são o id de um colaborador e os values são os projetos a que esse colaborador está associado
```

Função gera_slots(d, h_min, h_max):

```
d - quantidade de dias que queremos gerar
h_min - hora mínima de agendamento duma reunião
h_max - hora máxima de agendamento duma reunião
```

Esta função gera slots para o agendamento de reuniões.

In [1]:
```python
def gera_slots(d, h_min, h_max):
    slots = []
    for i in range(d):
        for j in range(h_min, h_max):
            slots.append((i,j))
    return slots
```

Função gera_disponibilidade(C,T):

```
C - Número de colaboradores
T - Lista com slots
```

Esta função gera de forma aleatória a disponibilidade dos colaboradores.

In [2]:
```python
def gera_disponibilidade(C,T):
    disp = {}
    for c in range(C):
        disp[c] = {}
        for t in T:
            disp[c][t] = randint(0,1)
    return disp
```

Função gera_projects_in(CP, C):

```
CP - Dicionário com os colaboradores por projeto
C - Número de colaboradores
```

Esta função gera um dicionário que diz se o colaborador C está associado ao projeto P.

In [3]:
```python
def gera_projects_in(CP, C):
    projects_in = {}

    for c in range(C):
        projects_in[c] = {}
        for p in CP:
            if c in CP[p]:
                projects_in[c][p] = 1
            else:
                projects_in[c][p] = 0



    return projects_in
```

Função print_disponibilidade(disp, hMin, hMax, dMax):

```
disp - Matriz de disponibilidade
hMin - hora mínima
hMax - hora máxima
dMax - dias máximos
```

Esta função dá print da disponibilidade de cada colaborador

In [4]:
```python
from tabulate import tabulate

def print_disponibilidade(disp, hMin, hMax, dMax):
    for c in disp:
        print('\nDisponibilidade do colaborador '+str(c))

        table = [[x] for x in range(hMin, hMax)]
        fst_line = ["Slots"]
        for t in T:
            if ("Dia " + str(t[0])) not in fst_line:
                fst_line.append("Dia " + str(t[0]))
        table.insert(0,fst_line)

        for d in range(0,dMax):
            for t in range(hMin, hMax):
                table[t+1-hMin].insert(d+1, "")
                if (disp[c][(d,t)]) == 0:
                    table[t+1-hMin][d+1] = "X"

        print(tabulate(table))
```

Função print_table(hMin, hMax, dMax, A, R):

```
hMin - hora mínima
hMax - hora máxima
dMax - dias máximos
A - Matriz de alocação de reuniões
R - Matriz de alocação de colaboradores a reuniões
```

Esta função dá print do resultado devolvido pelo solver.

In [5]:
```python
from tabulate import tabulate

def print_table(hMin, hMax, dMax, A, R):
    table = [[x] for x in range(hMin, hMax)]
    fst_line = ["Slots"]
    for t in T:
        if ("Dia " + str(t[0])) not in fst_line:
            fst_line.append("Dia " + str(t[0]))
    table.insert(0,fst_line)

    for d in range(0,dMax):
        for t in range(hMin, hMax):
            table[t+1-hMin].insert(d+1, "")
            for p in range(0, P):
                for s in range(0, S):
                    if int(A[p][s][(d,t)].solution_value()) == 1:
                        table[t+1-hMin][d+1] += ("\nProj %i - S %i\nCols: " % (p,s))
                        for c in range(0, C):
                            if int(R[c][p][s][(d,t)].solution_value()) == 1:
                                table[t+1-hMin][d+1] += f"{c}  "
                        table[t+1-hMin][d+1] = table[t+1-hMin][d+1][:-2]
    print(tabulate(table))
```

Matriz de alocação de reuniões

A matriz A serve para alocar reuniões de projetos p em salas s no slot t tem-se então

$$\forall_{p<P}.\forall_{s<S}.\forall_{t<T}.\ A_{p,s,t} == 1$$

se e só se existe uma reunião p na sala s no slot t

Matriz de alocação de colaboradores a uma reunião

A matriz R serve para alocar colaboradores a reuniões tem-se então

$$\forall_{c<C}\forall_{p<P}.\forall_{s<S}.\forall_{t<T}.\ R_{c,p,s,t} == 1$$

se e só se um colaborador tem uma reunião do projeto p na sala s no slot t

Restrições

```
1 - Disponibilidade dos colaboradores
```

$$\forall_{c<C}\forall_{p<P}\forall_{s<S}\forall_{t<T}\ R_{c,p,s,t} \le disp_{c,t}$$

Com esta restrição garantimos que só é alocada uma reunião a um colaborador C, num projeto P, numa sala S, num slot T caso o colaborador C esteja disponível.

```
2 - Cada projeto tem X ou mais reuniões semanais
```

$$\forall_{p<P}\sum_{s<S,t<T} A_{p,s,t} \ge X$$

Com esta restrição garantimos que um projeto P tem **pelo menos** X reuniões semanais.

```
3 - Não pode haver mais do que 1 reunião numa sala num slot
```

$$\forall_{s<S}\forall_{t<T}\sum_{p<P} A_{p,s,t} \le 1$$

Com esta restrição garantimos que numa sala S, num slot T, apenas está alocada uma ou nenhuma reunião.

```
4 - O colaborador só pode ir à reunião se tiver alocado ao projeto
```

$$\forall_{p<P}\forall_{c<C}\forall_{s<S}\forall_{t<T}\ R_{c,p,s,t} \le A_{p,s,t} \times projectsIn_{c,p}$$

Com esta restrição garantimos que um colaborador C só será alocado a uma reunião num projeto P, numa sala S, num slot T caso este esteja alocado ao projeto P correspondente.

```
5 - Um colaborador só pode ser alocado a uma reunião se esta estiver marcada
```

$$\forall_{t<T}\forall_{s<T}\forall_{p<P}\forall_{c<C}\ R_{c,p,s,t} \le A_{p,s,t}$$

Com esta restrição garantimos que um colaborador C só será alocado a uma reunião num slot T, numa sala S, a um projeto P caso a reunião tenha sido marcada.

```
6 - O líder tem que ir a todas as reuniões do projeto no qual é líder
```

$$\forall_{p<P,s<S,t<T} R_{LP_p,p,s,t} = A_{p,s,t}$$

Com esta restrição garantimos que qualquer líder de um projeto P tem de estar presente em todas as reuniões do projeto P numa sala S, num slot T.

```
7 - A assiduidade a cada reunião tem de ser superior a 50%
CPp dá-nos a lista com os colaboradores do projeto P
```

$$\forall_{p<P,s<S,t<T} \frac{\sum_{c<C} R_{c,p,s,t}}{len(CP_p)}$$

Com esta restrição garantimos que para qualquer projeto P, numa sala S, num slot T, a assiduidade é de, pelo menos, 50% face ao número de colaboradores existentes no projeto P.

```
8 - Um colaborador não pode estar em duas reuniões em simultâneo
```

$$\forall_{c<C,t<T}\sum_{s<S,p<P} R_{c,p,s,t} \le 1$$

Com esta restrição garantimos que para qualquer colaborador C, num slot T, este não está alocado a mais do que 1 sala S, em qualquer projeto P.

Maximizar o número de reuniões realizadas

Para maximizar o número de reuniões realizadas maximizamos o somatório de todas as reuniões alocadas na matriz de alocação de reuniões.

$$maximize \sum_{p<P,s<S,t<T} A_{p,s,t}$$

Minimizar o número médio de reuniões por participante

Para minimizar o número médio de reuniões por participantes, minimizamos o somatório das reuniões alocadas a colaboradores C na matriz R.

$$\forall_{c<C}\ minimize \sum_{p<P,s<S,t<T} R_{c,p,s,t}$$

In [6]:
```python
def horario(S, P, C, T, CP, projects_in, reunioes_semanais, disp):
    # Matriz de alocação de reuniões
    A = {}
    for p in range(P):
        A[p] = {}
        for s in range(S):
            A[p][s] = {}
            for t in T:
                A[p][s][t] = solver.BoolVar(f'A[{p}],[{s}],[{t}]')


    # Matriz de alocação de colaboradores a uma reunião
    R = {}
    for c in range(C):
        R[c] = {}
        for p in range(P):
            R[c][p] = {}
            for s in range(S):
                R[c][p][s] = {}
                for t in T:
                    R[c][p][s][t] = solver.BoolVar(f'R[{c},{p},{s},{t}]')

    # 1 - Disponibilidade dos colaboradores
    for c in range(C):
        for p in range(P):
            for s in range(S):
                for t in T:
                    solver.Add( R[c][p][s][t] <= disp[c][t])

    # 2 - Cada projeto tem x reuniões semanais
    for p in range(P):
        solver.Add( sum(A[p][s][t] for s in range(S) for t in T) >= reunioes_semanais)

    # 3 - Não pode haver mais do que 1 reunião numa sala num slot
    for s in range(S):
        for t in T:
            solver.Add( (sum(A[p][s][t] for p in range(P))) <= 1)

    # 4 - O colaborador só pode ir à reunião se tiver alocado ao projeto
    for p in range(P):
        for c in range(C):
            for s in range(S):
                for t in T:
                    solver.Add( R[c][p][s][t] <= A[p][s][t] * projects_in[c][p])

    # 5 - Um colaborador só pode estar em reunião se esta estiver marcada
    for t in T:
        for s in range(S):
            for p in range(P):
                for c in range(C):
                    solver.Add( R[c][p][s][t] <= A[p][s][t])

    # 6 - O Lider tem que ir a todas as reuniões do projeto no qual é líder
    for p in range(P):
        for s in range(S):
            for t in T:
                solver.Add( R[LP[p]][p][s][t] == A[p][s][t] )

    # 7 - A assiduidade a cada reunião tem de ser superior a 50%
    for p in range(P):
        for s in range(S):
            for t in T:
                solver.Add ( (sum(R[c][p][s][t] for c in range(C)) / len(CP[p])) >= 0.5*R[LP[p]][p][s][t] )

    # 8 - Um colaborador não pode estar em duas reuniões em simultâneo
    for c in range(C):
        for t in T:
            solver.Add( (sum(R[c][p][s][t] for s in range(S) for p in range(P))) <= 1)

    # Maximizar o número de reuniões realizadas
    solver.Maximize( sum(A[p][s][t] for p in range(P) for s in range(S) for t in T) )

    # Minimizar o número de reuniões por participante
    for c in range(C):
        solver.Minimize(sum(R[c][p][s][t] for p in range(P) for s in range(S) for t in T))

    status = solver.Solve()



    if status == pywraplp.Solver.OPTIMAL:
        print_table(T[0][1], T[-1][1] +1, 5, A, R)
    else:
        print("No solution found")
```

Exemplo 1

In [7]:
```python
from ortools.linear_solver import pywraplp
from random import randint

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

S = 2
P = 2
C = 4

# CP = {projeto: [colaboradores]}   CP -> colaboradores do projeto
CP = {
    0: [0,1,3],
    1: [0,2,3]
}


# projects_in = {colab: {project: 1(True) ou 0(False)}}  - dicionário que diz em que projetos o colaborador c está envolvido
projects_in = gera_projects_in(CP, C)
'''
projects_in = {
    0: {0:1,
        1:1},

    1: {0:1,
        1:0},

    2: {0:0,
        1:1},

    3: {0:1,
        1:1}
}
'''

# {projeto: lider}  LP -> Lider do projeto
LP = {
      0: 1,
      1: 2
}


T = gera_slots(5,9,18)
disp = gera_disponibilidade(C,T)
reunioes_semanais = 5


#print_disponibilidade(disp, 9, 18, 5)

horario(S, P, C, T, CP, projects_in, reunioes_semanais, disp)
```

```
-----  ------------  ------------  ------------  ------------  ------------
Slots  Dia 0         Dia 1         Dia 2         Dia 3         Dia 4
9
10     Proj 1 - S 0                              Proj 0 - S 0  Proj 0 - S 0
       Cols: 2  3                                Cols: 0  1    Cols: 0  1
11     Proj 1 - S 1                Proj 0 - S 1
       Cols: 0  2                  Cols: 0  1
12                                 Proj 0 - S 0
                                   Cols: 0  1
13                                                             Proj 1 - S 1
                                                               Cols: 0  2
14
15                   Proj 1 - S 1
                     Cols: 0  2
16
17                   Proj 1 - S 0                Proj 0 - S 0
                     Cols: 0  2                  Cols: 0  1
-----  ------------  ------------  ------------  ------------  ------------
```

Exemplo 2

In [8]:
```python
from ortools.linear_solver import pywraplp
from random import randint

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

S = 4
P = 5
C = 7

# CP = {projeto: [colaboradores]}   CP -> colaboradores do projeto
CP = {
    0: [0,1,2,3,6],
    1: [0,1,4,5],
    2: [2,3],
    3: [4,5,6],
    4: [1,3,4,5]
}

# projects_in = {colab: {project: 1(True) ou 0(False)}}  - dicionário que diz em que projetos o colaborador c está envolvido
projects_in = gera_projects_in(CP, C)


# {projeto: lider}  LP -> Lider do projeto
LP = {
      0: 1,
      1: 4,
      2: 2,
      3: 5,
      4: 3
}


reunioes_semanais = 10


T = gera_slots(5,9,19)
disp = gera_disponibilidade(C,T)

#print_disponibilidade(disp, T[0][0], T[-1][1]+1, 5)

horario(S, P, C, T, CP, projects_in, reunioes_semanais, disp)
```

```
-----  ------------  ------------  ------------  ------------  ------------
Slots  Dia 0         Dia 1         Dia 2         Dia 3         Dia 4
9      Proj 3 - S 0  Proj 4 - S 2  Proj 0 - S 0  Proj 1 - S 2  Proj 0 - S 0
       Cols: 4  5    Cols: 3  5    Cols: 1  2  3 Cols: 1  4    Cols: 0  1  3
                                   Proj 3 - S 3                Proj 2 - S 2
                                   Cols: 4  5                  Cols: 2
10     Proj 1 - S 1  Proj 4 - S 0  Proj 1 - S 0  Proj 4 - S 3  Proj 2 - S 2
       Cols: 4  5    Cols: 3  5    Cols: 1  4    Cols: 3  4    Cols: 2
                                   Proj 2 - S 2                Proj 3 - S 1
                                   Cols: 2                     Cols: 4  5
11     Proj 0 - S 3  Proj 0 - S 3                              Proj 4 - S 1
       Cols: 0  1  2 Cols: 0  1  3                             Cols: 1  3
                     Proj 2 - S 2
                     Cols: 2
12     Proj 4 - S 2                Proj 1 - S 2
       Cols: 3  5                  Cols: 0  4
                                   Proj 2 - S 1
                                   Cols: 2
13     Proj 0 - S 0  Proj 2 - S 2  Proj 1 - S 0
       Cols: 0  1  2 Cols: 2       Cols: 1  4
                     Proj 4 - S 3
                     Cols: 1  3
14                   Proj 3 - S 0  Proj 1 - S 0  Proj 2 - S 1  Proj 0 - S 3
                     Cols: 4  5    Cols: 4  5    Cols: 2       Cols: 0  1  2
15     Proj 1 - S 1  Proj 2 - S 1  Proj 4 - S 3  Proj 1 - S 0
       Cols: 0  4    Cols: 2       Cols: 3  5    Cols: 4  5
       Proj 4 - S 2
       Cols: 1  3
16     Proj 4 - S 0  Proj 2 - S 1  Proj 0 - S 1                Proj 1 - S 3
       Cols: 3  4    Cols: 2       Cols: 1  2  3               Cols: 0  4
                                   Proj 3 - S 2
                                   Cols: 4  5
17     Proj 4 - S 1                Proj 3 - S 1                Proj 0 - S 3
       Cols: 3  5                  Cols: 4  5                  Cols: 1  2  3
                                                               Proj 1 - S 1
                                                               Cols: 0  4
18     Proj 2 - S 3  Proj 0 - S 1  Proj 3 - S 3  Proj 0 - S 3  Proj 3 - S 3
       Cols: 2       Cols: 0  1  3 Cols: 4  5    Cols: 0  1  3 Cols: 4  5
                     Proj 3 - S 0                Proj 3 - S 2
                     Cols: 4  5                  Cols: 4  5
-----  ------------  ------------  ------------  ------------  ------------
```

Exemplo 3

In [9]:
```python
from ortools.linear_solver import pywraplp
from random import randint

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

S = 4
P = 5
C = 10

# CP = {projeto: [colaboradores]}   CP -> colaboradores do projeto
CP = {
    0: [0,1,2,3,6,8],
    1: [0,1,4,5,9],
    2: [2,3,7,8],
    3: [4,5,6,7,9],
    4: [1,3,4,5]
}

# projects_in = {colab: {project: 1(True) ou 0(False)}}  - dicionário que diz em que projetos o colaborador c está envolvido
projects_in = gera_projects_in(CP, C)


# {projeto: lider}  LP -> Lider do projeto
LP = {
      0: 1,
      1: 4,
      2: 2,
      3: 5,
      4: 3
}


reunioes_semanais = 10


T = gera_slots(5,9,19)
disp = gera_disponibilidade(C,T)

#print_disponibilidade(disp, T[0][0], T[-1][1]+1, 5)

horario(S, P, C, T, CP, projects_in, reunioes_semanais, disp)
```

```
-----  ----------------  ------------  ------------  ----------------  ------------
Slots  Dia 0             Dia 1         Dia 2         Dia 3             Dia 4
9                        Proj 3 - S 1  Proj 4 - S 3                    Proj 1 - S 2
                         Cols: 4  5  6 Cols: 3  4                      Cols: 0  4  5
                                                                       Proj 2 - S 1
                                                                       Cols: 2  3
10     Proj 2 - S 1      Proj 0 - S 2  Proj 0 - S 2  Proj 1 - S 2      Proj 4 - S 3
       Cols: 2  7        Cols: 1  2  6 Cols: 0  1  2 Cols: 0  1  4     Cols: 3  5
       Proj 3 - S 2                                  Proj 2 - S 1
       Cols: 4  5  6                                 Cols: 2  3
                                                     Proj 3 - S 0
                                                     Cols: 5  7  9
11     Proj 1 - S 3      Proj 0 - S 2                Proj 3 - S 2      Proj 1 - S 3
       Cols: 0  1  4     Cols: 0  1  2               Cols: 4  5  6     Cols: 1  4  5
                         Proj 3 - S 1
                         Cols: 5  7  9
12     Proj 1 - S 0      Proj 2 - S 3  Proj 0 - S 0  Proj 2 - S 3
       Cols: 0  1  4     Cols: 2  8    Cols: 0  1  8 Cols: 2  7
                                                     Proj 4 - S 0
                                                     Cols: 3  4
13                       Proj 0 - S 0  Proj 2 - S 0  Proj 0 - S 1
                         Cols: 1  3  6 Cols: 2  8    Cols: 0  1  2  3
                                                     Proj 3 - S 0
                                                     Cols: 5  6  9
14                       Proj 4 - S 0  Proj 3 - S 2  Proj 4 - S 2      Proj 1 - S 0
                         Cols: 3  4    Cols: 4  5  7 Cols: 3  4        Cols: 0  1  4
                                                                       Proj 2 - S 1
                                                                       Cols: 2  8
15     Proj 2 - S 1      Proj 1 - S 1  Proj 4 - S 3  Proj 1 - S 2      Proj 4 - S 1
       Cols: 2  3        Cols: 0  1  4 Cols: 3  5    Cols: 4  5  9     Cols: 1  3
                         Proj 3 - S 3
                         Cols: 5  6  9
16     Proj 4 - S 1                    Proj 2 - S 1                    Proj 3 - S 0
       Cols: 1  3                      Cols: 2  7                      Cols: 5  6  7
                                                                       Proj 4 - S 1
                                                                       Cols: 3  4
17                       Proj 0 - S 3                Proj 0 - S 2      Proj 1 - S 0
                         Cols: 1  3  8               Cols: 1  2  6     Cols: 0  4  5
                                                     Proj 4 - S 0
                                                     Cols: 3  4
18     Proj 0 - S 3      Proj 0 - S 0                Proj 3 - S 3      Proj 1 - S 3
       Cols: 1  2  6  8  Cols: 1  6  8               Cols: 4  5  6     Cols: 1  4  5
                                                                       Proj 2 - S 1
                                                                       Cols: 2  7
-----  ----------------  ------------  ------------  ----------------  ------------
```

Exemplo 4

In [10]:
```python
from ortools.linear_solver import pywraplp
from random import randint

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

S = 4
P = 7
C = 15

# CP = {projeto: [colaboradores]}   CP -> colaboradores do projeto
CP = {
    0: [0,1,2,3,6,8],
    1: [0,1,4,5,9],
    2: [2,3,7,8,10,11],
    3: [4,5,6,7,9],
    4: [1,3,4,5,12,13,14],
    5: [1,2,8,12,13],
    6: [0,5,9,10,12,14]
}

# projects_in = {colab: {project: 1(True) ou 0(False)}}  - dicionário que diz em que projetos o colaborador c está envolvido
projects_in = gera_projects_in(CP, C)


# {projeto: lider}  LP -> Lider do projeto
LP = {
      0: 1,
      1: 4,
      2: 2,
      3: 5,
      4: 3,
      5: 12,
      6: 9
}


reunioes_semanais = 10


T = gera_slots(5,9,19)
disp = gera_disponibilidade(C,T)

#print_disponibilidade(disp, T[0][0], T[-1][1]+1, 5)

horario(S, P, C, T, CP, projects_in, reunioes_semanais, disp)
```

```
-----  ----------------  ----------------  -----------------  ------------------  ----------------
Slots  Dia 0             Dia 1             Dia 2              Dia 3               Dia 4
9                        Proj 3 - S 0      Proj 4 - S 3       Proj 1 - S 1        Proj 2 - S 0
                         Cols: 5  6  7     Cols: 1  3  12  13 Cols: 0  4  9       Cols: 2  7  10
                                                              Proj 2 - S 2        Proj 6 - S 2
                                                              Cols: 2  8  10      Cols: 0  5  9
                                                              Proj 4 - S 0
                                                              Cols: 3  5  12  13
10     Proj 1 - S 0                        Proj 3 - S 0       Proj 0 - S 0        Proj 5 - S 0
       Cols: 0  1  4                       Cols: 4  5  6      Cols: 1  6  8       Cols: 8  12  13
11     Proj 0 - S 0      Proj 0 - S 3      Proj 1 - S 1       Proj 0 - S 0        Proj 6 - S 3
       Cols: 1  3  8     Cols: 1  2  6     Cols: 1  4  5      Cols: 1  2  3  6  8 Cols: 0  9  10
                                           Proj 5 - S 3       Proj 1 - S 3
                                           Cols: 2  8  12     Cols: 0  4  5
                                                              Proj 6 - S 1
                                                              Cols: 9  10  12
12     Proj 0 - S 1      Proj 3 - S 0      Proj 5 - S 0                           Proj 0 - S 2
       Cols: 0  1  3     Cols: 4  5  6     Cols: 2  12  13                        Cols: 0  1  8
       Proj 3 - S 0      Proj 6 - S 3                                             Proj 2 - S 3
       Cols: 5  6  7     Cols: 9  10  12                                          Cols: 2  7  10
13     Proj 0 - S 1                        Proj 4 - S 3       Proj 0 - S 0        Proj 4 - S 0
       Cols: 0  1  6                       Cols: 1  3  5  13  Cols: 0  1  3       Cols: 3  4  5  13
       Proj 2 - S 0                                           Proj 2 - S 2
       Cols: 2  8  11                                         Cols: 2  8  11
       Proj 6 - S 3
       Cols: 5  9  12
14     Proj 4 - S 3      Proj 6 - S 1      Proj 1 - S 1       Proj 5 - S 0        Proj 3 - S 2
       Cols: 1  3  4  13 Cols: 5  9  10    Cols: 0  1  4      Cols: 2  8  12      Cols: 4  5  6
       Proj 5 - S 2                        Proj 2 - S 0                           Proj 5 - S 1
       Cols: 2  8  12                      Cols: 2  3  7                          Cols: 1  2  12
       Proj 6 - S 0
       Cols: 5  9  10
15     Proj 4 - S 1      Proj 4 - S 2                         Proj 3 - S 3        Proj 1 - S 0
       Cols: 3  4  5  13 Cols: 3  4  5  13                    Cols: 5  6  7       Cols: 0  1  4
                                                              Proj 5 - S 0
                                                              Cols: 1  12  13
16                       Proj 0 - S 0      Proj 3 - S 1       Proj 0 - S 1        Proj 1 - S 2
                         Cols: 1  2  3     Cols: 5  6  7      Cols: 1  2  6       Cols: 1  4  9
                         Proj 3 - S 2                         Proj 4 - S 0
                         Cols: 5  6  9                        Cols: 3  4  12  13
                                                              Proj 6 - S 2
                                                              Cols: 0  9  10
17     Proj 2 - S 1      Proj 1 - S 1      Proj 1 - S 0       Proj 2 - S 1        Proj 3 - S 0
       Cols: 2  3  8     Cols: 0  4  9     Cols: 0  4  9      Cols: 2  3  8       Cols: 5  6  9
                         Proj 4 - S 2      Proj 2 - S 3       Proj 6 - S 3        Proj 5 - S 1
                         Cols: 1  3  5  12 Cols: 2  7  8      Cols: 0  9  12      Cols: 2  8  12
18     Proj 3 - S 3      Proj 6 - S 0      Proj 4 - S 0       Proj 1 - S 2        Proj 5 - S 0
       Cols: 5  7  9     Cols: 0  9  12    Cols: 3  4  5  12  Cols: 1  4  5  9    Cols: 1  12  13
       Proj 5 - S 2                                           Proj 2 - S 1
       Cols: 2  8  12                                         Cols: 2  8  10
-----  ----------------  ----------------  -----------------  ------------------  ----------------
```

In [ ]: