In [3]:
```
import numpy as np
from gurobipy import *
```

In [4]:
```
set_num = 5
total_num = 8
S = [{0,1,7},{0,4,5},{2,5},{1,6,7},{3,4}]
Set_Contained = [{0,1},{0,3},{2},{4},{1,4},{1,2},{3},{0,3}]
```

In [5]:
```
WW = Model()
Choose = WW.addVars(set_num, vtype=GRB.BINARY, name = "Choosen_set")
WW.setObjective(quicksum(Choose[i] for i in range(set_num)), GRB.MINIMIZE)
WW.addConstrs(quicksum(Choose[i] for i in Set_Contained[j])>=1 for j in range(total_num))
```

```
Restricted license - for non-production use only - expires 2022-01-13
```

Out[5]:
```
{0: <gurobi.Constr *Awaiting Model Update*>,
 1: <gurobi.Constr *Awaiting Model Update*>,
 2: <gurobi.Constr *Awaiting Model Update*>,
 3: <gurobi.Constr *Awaiting Model Update*>,
 4: <gurobi.Constr *Awaiting Model Update*>,
 5: <gurobi.Constr *Awaiting Model Update*>,
 6: <gurobi.Constr *Awaiting Model Update*>,
 7: <gurobi.Constr *Awaiting Model Update*>}
```

```
WW.optimize()
WW.printAttr('X')
```

```
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (win64)
Thread count: 4 physical cores, 8 logical processors, using up to 8 threads
Optimize a model with 8 rows, 5 columns and 13 nonzeros
Model fingerprint: 0x74c2bfc5
Variable types: 0 continuous, 5 integer (5 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [1e+00, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+00]
Found heuristic solution: objective 4.0000000
Presolve removed 8 rows and 5 columns
Presolve time: 0.01s
Presolve: All rows and columns removed

Explored 0 nodes (0 simplex iterations) in 0.03 seconds
Thread count was 1 (of 8 available processors)

Solution count 1: 4

Optimal solution found (tolerance 1.00e-04)
Best objective 4.000000000000e+00, best bound 4.000000000000e+00, gap 0.0000%

    Variable            X
-------------------------
Choosen_set[1]            1
Choosen_set[2]            1
Choosen_set[3]            1
Choosen_set[4]            1
```

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

P1.

Let U_e represents all the sets which contain element e.

Decision variable: C_i, which is a binary number indicating whether S_i is chosen

objective: $\min \sum_{S_i \in S} C_i$

Subject to: $\sum_{S_i \in U_e} C_i \geq 1 \quad \forall e \in E$

$C_i \in \{0,1\}$

P2.

Assume a strategy $S = \{x_1, x_2, \ldots, x_n\}$

a modified strategy $S' = \{x_1, x_2, \ldots x_i', \ldots x_j', \ldots, x_n\}$

And $x_i > x_i'$, $x_j < x_j'$, $\frac{v_i}{s_i} > \frac{v_j}{s_j}$

We have: $\begin{cases} v_i(x_i - x_i') < v_j(x_j' - x_j) \\ s_i(x_i - x_i') \geq s_j(x_j' - x_j) \end{cases}$ if S' is optimal then S

$\Rightarrow \frac{v_i}{s_i} < \frac{v_j}{s_j} \Rightarrow$ contradiction.

Here, we prove that any strategy which decrease former weights and increase later weights will lead to a worse result.

So, the greedy rule is the best solution, since it is:

$\{x_1, x_2, \ldots x_n\} \quad \begin{cases} 1 & i<k \\ \alpha & i=k \\ 0 & i>k \end{cases}$