Altay Problem

In [27]:
```
import sys
import pandas as pd
import numpy as np

from gamspy import (
    Container,Set,Alias,Parameter,Variable,Equation,Model,Problem,Sense,Opti
    Domain,Number,Sum,Product,Smax,Smin,Ord,Card,SpecialValues,
    ModelStatus,SolveStatus,
)

options = Options(time_limit=10000000,relative_optimality_gap=0.)

m = Container(options=options)

digits = m.addSet('digits', records=["0", "1", "2", "3", "4", "5", "6", "7",
i = m.addAlias('i', digits)
j = m.addAlias('j', digits)

numbers = Parameter(m, 'number_value', domain=[digits], records = [("0", 0),
                                                                   ("1", 1),
                                                                   ("2", 2),
                                                                   ("3", 3),
                                                                   ("4", 4),
                                                                   ("5", 5),
                                                                   ("6", 6),
                                                                   ("7", 7),
                                                                   ("8", 8),
                                                                   ("9", 9)]


touching_matrix = m.addParameter('touching', domain=[i, j], records = np.arr
                                                                      [1,
                                                                      [4,
                                                                      [5,
                                                                      [5,
                                                                      [4,
                                                                      [4,
                                                                      [5,
                                                                      [5,
                                                                      [5,


x = m.addVariable('x', 'binary', domain=[i,j])
y = m.addVariable('y', 'positive', domain=[digits])

eq1 = m.addEquation('eq1', domain=j)
eq1[j] = Sum(i, x[i, j]) <= 1

eq2 = m.addEquation('eq2', domain=i)
eq2[i] = Sum(j, x[i, j]) <= 1

eq3 = m.addEquation('eq3')
```

```
eq3[:] = Sum([i, j], x[i, j]) == 9

elim_circle = m.addEquation('elim_circle', domain=[i, j])
elim_circle[i, j] = y[i] - y[j] + 10 * x[i, j] <= 9

p_const = m.addEquation('p_const', domain=i)
p_const[i] = y[i] >= 0

altay = m.addModel('altay',
    equations=m.getEquations(),
    problem=Problem.MIP,
    sense=Sense.MAX,
    objective=Sum([i, j], (numbers[i] + numbers[j]) * (touching_matrix[i, j]
)
```

In [28]:
```
# Definition of n and model here
altay.solve()
```

Out[28]:

	Solver Status	Model Status	Objective	Num of Equations	Num of Variables	Model Type	Solver	Solve Tim
0	Normal	OptimalGlobal	352	132	111	MIP	CPLEX	0.11

In [29]:
```
elements = x.records['level'] > 0
display(x.records.loc[elements])
print(f'positions are:')
display(y.records)
```

	i	j	level	marginal	lower	upper	scale
7	0	7	1.0	14.0	0.0	1.0	1.0
21	2	1	1.0	6.0	0.0	1.0	1.0
34	3	4	1.0	21.0	0.0	1.0	1.0
49	4	9	1.0	52.0	0.0	1.0	1.0
53	5	3	1.0	24.0	0.0	1.0	1.0
62	6	2	1.0	32.0	0.0	1.0	1.0
75	7	5	1.0	48.0	0.0	1.0	1.0
86	8	6	1.0	70.0	0.0	1.0	1.0
98	9	8	1.0	85.0	0.0	1.0	1.0

```
positions are:
```

	digits	level	marginal	lower	upper	scale
0	0	0.0	-0.0	0.0	inf	1.0
1	1	9.0	0.0	0.0	inf	1.0
2	2	8.0	0.0	0.0	inf	1.0
3	3	3.0	0.0	0.0	inf	1.0
4	4	4.0	0.0	0.0	inf	1.0
5	5	2.0	0.0	0.0	inf	1.0
6	6	7.0	0.0	0.0	inf	1.0
7	7	1.0	0.0	0.0	inf	1.0
8	8	6.0	0.0	0.0	inf	1.0
9	9	5.0	0.0	0.0	inf	1.0

In [30]:
```
# pi is inverse permutation of z: ordering
# update code below to be consistent with you variables
zvals = y.toDense()
pi = np.zeros(10,)

for k in range(10):
    pi[int(zvals[k])] = k
print(f'ordering is:')
display(pi)
```

```
ordering is:
array([0., 7., 5., 3., 4., 9., 8., 6., 2., 1.])
```

In [ ]:
```

```