Term Project

IE400 - Principle of Engineering Management

Göktuğ Gürbüztürk - 21702383

Ege Şahin - 21702300

Bulut Gözübüyük - 21702771

Group No: 2

Problem Description

In this problem, we are supposed to position the binary string with length l on an n by n grid where each character of the string should be on a node of the grid. While doing this the following three constraints should be satisfied.

1) Each node of the grid can host at most one character of the string
2) Each character of the string can be assigned to exactly one node of the grid
3) Two consecutive characters on the string should be assigned to the neighboring nodes of the grid.

If there are two ‘1’s that are assigned to neighboring nodes of the grid and these characters are not consecutive on the string, that edge is called as contact. According to the problem, we are supposed to maximize the number of contacts in the grid after positioning.

Part A

- **Model**
  - **Parameters**

$$N_{ij} : n^2 \times n^2 \text{ matrix that represents the neighbors of each node} \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2$$

$$C_k : \text{binary value in the } k^{\text{th}} \text{ position of string} \quad k = 1, \ldots, l$$

  - **Decision Variables**

$$G_{ik} = \begin{cases} 1, & \text{if the } k^{\text{th}} \text{ char of string is assigned to node i,} \\ 0, & \text{otherwise.} \end{cases} \qquad i = 1, \ldots, n^2 \quad k = 1, \ldots, l$$

$$I_i = \begin{cases} 1, & \text{if the node i keeps '1' binary value} \\ 0, & \text{otherwise.} \end{cases} \qquad i = 1, \ldots, n^2$$

$$E_{ij} = \begin{cases} 1, & \text{if both node i and node j keep '1' binary value } (\mathrm{I}_i = I_j = 1), \\ 0, & \text{otherwise.} \end{cases} \qquad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2$$

- **Objective Function**

$$\max \frac{\sum_{i=1}^{n^2} \sum_{j=1}^{n^2} E_{ij} N_{ij}}{2} - \left(\sum_{k=1}^{l-1} C_k C_{k+1}\right)$$

- **Constraints**

$$\sum_{i=1}^{n^2} G_{ik} = 1 \quad k = 1, \ldots, l \tag{1}$$

$$\sum_{k=1}^{l} G_{ik} \leq 1 \quad i = 1, \ldots, n^2 \tag{2}$$

$$G_{ik} - \sum_{j=1}^{n^2} N_{ij} G_{j(k-1)} = 0 \quad k = 2, \ldots, l \quad i = 1, \ldots, n^2 \tag{3}$$

$$I_i = \sum_{k=1}^{l} G_{ik} C_k \quad i = 1, \ldots, n^2 \tag{4}$$

$$E_{ij} \leq I_i \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2 \tag{5}$$

$$E_{ij} \leq I_j \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2 \tag{6}$$

$$E_{ij} \geq I_i + I_j - 1 \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2 \tag{7}$$

$$I_i \in \{0, 1\} \qquad E_{ij} \in \{0, 1\} \qquad G_{ik} \in \{0, 1\}$$

- **Explanation of the Model (Constraints, Objective Function, Parameters, Decision variables)**

At first, the nodes on the n x n grid are represented with the indexes from 1 to n^2. In the model there are two different parameters. Parameter Nij is an $n^2 x\, n^2$ matrix that represents the neighbors of each node on the grid. If the jth node is a neighbor of the ith node, then N_{ij} is equal to 1 otherwise it is equal to 0 which means that they are not adjacent on the grid. The second parameter C_k is the binary value of the character on the kth position of the given string. Each character in the string is indexed from 1 to length of the string(l).

In the model there are three different decision variables. The decision variable G_{ik} is used to decide on the assignment relationship between characters and the nodes on the grid. If the kth char of the string is assigned to ith node on the grid, G_{ik} will be equal to 1 otherwise it will be equal to 0. The second decision variable I_i is used to decide which nodes host characters with '1' binary value. If the char on the node number i is 1, then the value of I_i is also 1. I_i is 0 if the ith node contains 0 as character or it contains no characters.The last decision variable is E_{ik} which is used to decide if both ith and jth node host characters with '1' binary value. It is equal to one if both node i and node j keeps '1'. It is used to find the adjacent nodes with 1 binary value by combining them with N_{ij} parameters.

In the model there are three main constraints as specified in the problem description. First three constraints are specified by the problem. They are numbered as 1, 2 and 3 in our model. The first constraint is to ensure that each node of the grid can host at most one character of the string. The second constraint implies that each character of the string must be assigned to exactly one node of the grid. The third constraint is to ensure that the two consecutive characters on the string must be positioned to the neighboring nodes of the grid.

Also, there are some other constraints in our model (numbered as 4, 5, 6 and 7). These constraints are used to limit the values of the decision variables Eik and Ii and explain the relation between E_{ik} and I_i with parameters and other variables. Constraint 4 implies that the kth character of the string is assigned to ith node and its value is 1. Constraints 5, 6 and 7 are the linearized form of the '$E_{ij} = I_i * I_j$' which implies that both node i and node j host '1' binary value.

In the objective function, it is aimed to find the maximum number of contacts on the grid. First part of the objective function, represented with a nested sum, is equal to the total number of edges between nodes with '1' binary values and the second part gives the number of consecutive 1's in the string. When the difference of them is taken, we get the number of contacts on the grid.

- **Results (Objective Values of the models for strings S1, S2, S3 etc)**
    - **String S1, n = 4:**

```
Result - Optimal solution found

Objective value:                2.00000000
```

  - **String S2, n = 5:**

```
Result - Optimal solution found

Objective value:                4.00000000
```

  - **String S3 n = 6:**

```
Result - Optimal solution found

Objective value:                8.00000000
```

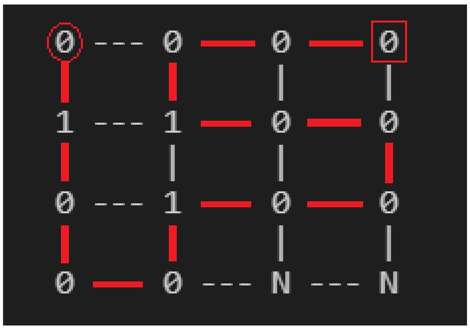
- **Illustration of each part on the given grids.**
    - **String S1, n = 4:**

This first picture illustrates where the characters of the string are placed on the grid. It can be seen that the first character is placed on the upper left corner and the second character is beneath it and it continues in the order until the whole string is placed.

```
1    12   13   14
2    11   10   9
3    6    7    8
4    5    N    N
```

The left grid here is the simple output and the right one is the grid which shows where the string is started on the grid where it ends on the grid and the string's path from starting node to ending node. The node inside the circle is the starting node and the node in square is the ending node.

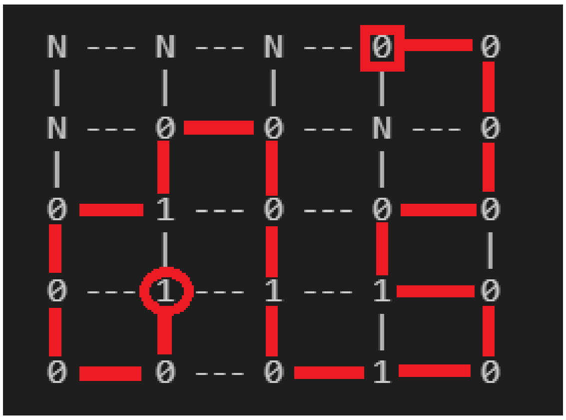
```
0 --- 0 --- 0 --- 0
|     |     |     |
1 --- 1 --- 0 --- 0
|     |     |     |
0 --- 1 --- 0 --- 0
|     |     |     |
0 --- 0 --- N --- N
```



- **String S2, n = 5:**

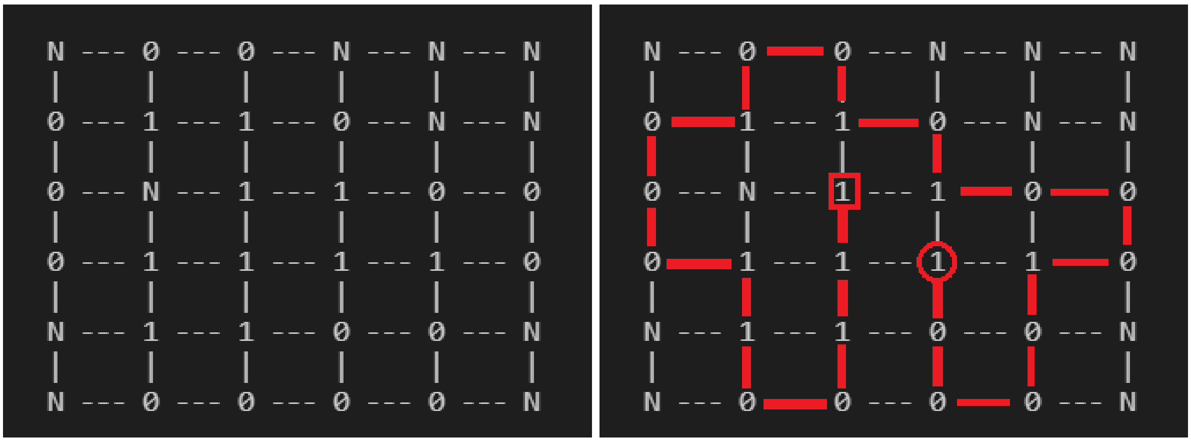
```
N    N    N    20   19
N    7    8    N    18
5    6    9    16   17
4    1    10   15   14
3    2    11   12   13
```

```
N --- N --- N --- 0 --- 0
|     |     |     |     |
N --- 0 --- 0 --- N --- 0
|     |     |     |     |
0 --- 1 --- 0 --- 0 --- 0
|     |     |     |     |
0 --- 1 --- 1 --- 1 --- 0
|     |     |     |     |
0 --- 0 --- 0 --- 1 --- 0
```



- **String S3, n = 6:**

```
N    14   13   N    N    N
16   15   12   11   N    N
17   N    25   10   9    8
18   19   24   1    6    7
N    20   23   2    5    N
N    21   22   3    4    N
```



Part B

- **Modified Model**

In part B we used the previous model and added one more parameter, two more decision variables and five more constraints to the previous model.

- **Model of part A:**

$$N_{ij} : n^2 \times n^2 \text{ matrix that represents the neighbors of each node} \quad i = 1, \dots, n^2 \quad j = 1, \dots, n^2$$

$$C_k : \text{binary value in the } k^{\text{th}} \text{ position of string} \quad k = 1, \dots, l$$

$$G_{ik} = \begin{cases} 1, & \text{if the } k^{\text{th}} \text{ char of string is assigned to node i,} \\ 0, & \text{otherwise.} \end{cases} \quad i = 1, \dots, n^2 \quad k = 1, \dots, l$$

$$I_i = \begin{cases} 1, & \text{if the node i keeps '1' binary value} \\ 0, & \text{otherwise.} \end{cases} \quad i = 1, \dots, n^2$$

$$E_{ij} = \begin{cases} 1, & \text{if both node i and node j keep '1' binary value } (\mathrm{I}_i = I_j = 1), \\ 0, & \text{otherwise.} \end{cases} \quad i = 1, \dots, n^2 \quad j = 1, \dots, n^2$$

$$\max \frac{\sum_{i=1}^{n^2}\sum_{j=1}^{n^2} E_{ij}N_{ij}}{2} - (\sum_{k=1}^{l-1} C_k C_{k+1})$$

$$\textstyle\sum_{i=1}^{n^2} G_{ik} = 1 \quad k = 1, \ldots, l \tag{1}$$

$$\textstyle\sum_{k=1}^{l} G_{ik} \le 1 \quad i = 1, \ldots, n^2 \tag{2}$$

$$G_{ik} - \textstyle\sum_{j=1}^{n^2} N_{ij}G_{j(k-1)} = 0 \quad k = 2, \ldots, l \quad i = 1, \ldots, n^2 \tag{3}$$

$$I_i = \textstyle\sum_{k=1}^{l} G_{ik}C_k \quad i = 1, \ldots, n^2 \tag{4}$$

$$E_{ij} \le I_i \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2 \tag{5}$$

$$E_{ij} \le I_j \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2 \tag{6}$$

$$E_{ij} \ge I_i + I_j - 1 \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2 \tag{7}$$

$$I_i \in \{0, 1\} \qquad E_{ij} \in \{0, 1\} \qquad G_{ik} \in \{0, 1\}$$

- **Additions of part B:**
  - **Parameter:**

$$C'_k : \text{binary value in the } k^{\text{th}} \text{ position of inverted string} \quad k = 1, \ldots, l$$

  - **Decision Variables:**

$$I'_i = \begin{cases} 1, & \text{if the } i^{\text{th}} \text{ node contains } k^{\text{th}} \text{ char of inverted string} \\ 0, & \text{otherwise.} \end{cases} \qquad i = 1, \ldots, n^2$$

$$E'_{ij} = \begin{cases} 1, & \text{if } \mathrm{I'}_i = I'_j = 1, \\ 0, & \text{otherwise.} \end{cases} \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2$$

- **Constraints:**

$$I'_i = \sum_{k=1}^{l} G_{ik} C'_k \quad i = 1, \ldots, n^2 \tag{8}$$

$$E'_{ij} \leq I'_i \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2 \tag{9}$$

$$E'_{ij} \leq I'_j \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2 \tag{10}$$

$$E'_{ij} \geq I'_i + I'_j - 1 \quad i = 1, \ldots, n^2 \quad j = 1, \ldots, n^2 \tag{11}$$

$$\frac{\sum_{i=1}^{n^2} \sum_{j=1}^{n^2} E'_{ij} N_{ij}}{2} - \left(\sum_{k=1}^{l-1} C'_k C'_{k+1}\right) \tag{12}$$

$$I'_i \in \{0, 1\} \qquad E'_{ij} \in \{0, 1\}$$

- **Explanation of the Model (Constraints, Objective Function, Parameters, Decision variables)**

In part B, we add a new parameter C'_k. This parameter is similar to C_k of the part A. C'_k is the binary value of the character on the kth position of the inverted string (If the original string is (1,0,0,1) inverted string is (0,1,1,0)). Similar to C_k, each character in the inverted string is indexed from 1 to length of the string(l).

As decision variables, I'_i and E'_{ij} are added. I'_i is used to decide which nodes host characters of inverted string with ‘1’ binary value. If the character of the inverted string on the node number i is 1, then the value of I'_i is also 1. I'_i

is 0 if the ith node contains 0 as a character from the inverted string or it contains no characters. In other words, basically I'_i is 1 if the kth character of the original string is 0 and it is positioned on the ith node.The second decision variable is E'_{ij} which is used to decide if both ith and jth nodes host characters with ‘1’ binary value from inverted string. It is equal to one if both node i and node j keeps ‘1’ and ‘0’ otherwise.. In other words, E'_{ij} is 1 if both ith and jth nodes contain 0 binary values from the original string. The reason that relations between 0 binary values of the original string is modeled over 1 binary values of an inverted string is that it is easier to work with 1s instead of 0s.

The first constraint (constraint number 8) limits the values of I'_i and explains the relation between I'_i and other parameters and decision variables. The relation between E'_{ij}, I'_i and I'_j is $E'_{ij}= I'_i * I'_j$. Next three constraints (constraints numbered as 9, 10, 11) are the linearized version of $E'_{ij} = I'_i * I'_j$. The last constraint forbids non consecutive 0s of original string to be placed on adjacent nodes on the grid. The first nested sum calculates the two times the number of edges between 1s of inverted string on the grid. Since each edge is considered two times, this value is divided by two. This value is equal to the number of edges between 0s of the original string on the grid. Then the number of 1s that are consecutive on the inverted string is calculated. This value is equal to the number of consecutive 0s on the original string. At last, these two numbers must be equal to each other which means no 0s of original string which are not consecutive can be placed on adjacent nodes of the string.

- **Results (Objective Values of the models for strings S1, S2, S3 etc)**
    - **String S1, n = 4:**

```
Result - Problem proven infeasible

No feasible solution found
```

-
    - **String S1, n = 5:**

```
Result - Optimal solution found

Objective value:                1.00000000
```

- **String S2, n = 5:**

```
Result - Optimal solution found

Objective value:                2.00000000
```

- **String S3, n = 6**

```
Result - Optimal solution found

Objective value:                7.00000000
```

- **Illustration of each part on the given grids.**
  - **String S1, n = 4:**

There is no solution.

  - **String S1, n = 5:**

```
N     N     14    N     N
N     12    13    2     1
10    11    N     3     N
9     6     5     4     N
8     7     N     N     N
```

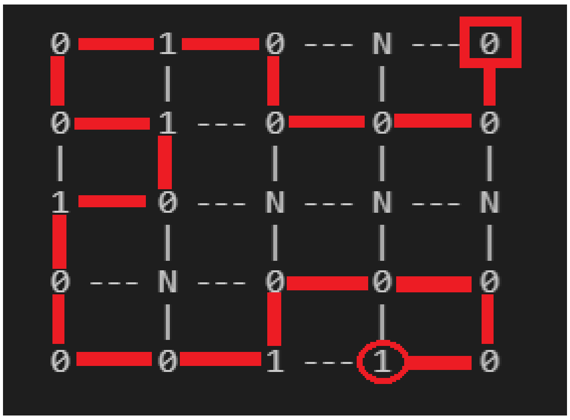
```
N --- N --- 0 --- N --- N
|     |     |     |     |
N --- 0 --- 0 --- 1 --- 0
|     |     |     |     |
0 --- 1 --- N --- 0 --- N
|     |     |     |     |
0 --- 1 --- 0 --- 0 --- N
|     |     |     |     |
0 --- 0 --- N --- N --- N
```

```
N --- N --- 0 --- N --- N
|     |           |     |
N --- 0     0 --- 1     0
|           |           |
0     1 --- N --- 0 --- N
      |     |           |
0 --- 1     0     0 --- N
            |     |     |
0     0 --- N --- N --- N
```

- **String S2, n = 5:**

```
14   15   16   N    20
13   12   17   18   19
10   11   N    N    N
9    N    5    4    3
8    7    6    1    2
```

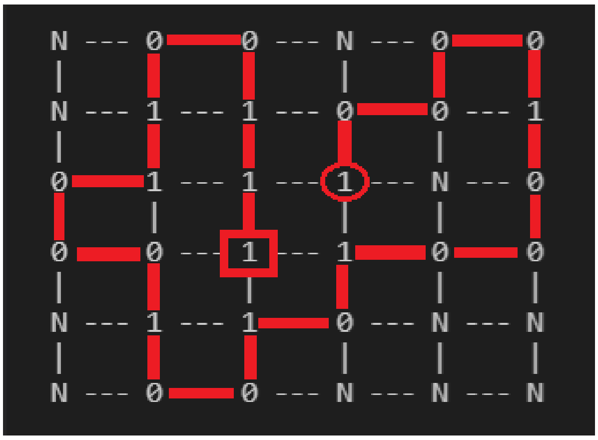
```
0 --- 1 --- 0 --- N --- 0
|     |     |     |     |
0 --- 1 --- 0 --- 0 --- 0
|     |     |     |     |
1 --- 0 --- N --- N --- N
|     |     |     |     |
0 --- N --- 0 --- 0 --- 0
|     |     |     |     |
0 --- 0 --- 1 --- 1 --- 0
```



- **String S3, n = 6:**

```
N    21   22   N    4    5
N    20   23   2    3    6
18   19   24   1    N    7
17   16   25   10   9    8
N    15   12   11   N    N
N    14   13   N    N    N
```

```
N --- 0 --- 0 --- N --- 0 --- 0
|     |     |     |     |     |
N --- 1 --- 1 --- 0 --- 0 --- 1
|     |     |     |     |     |
0 --- 1 --- 1 --- 1 --- N --- 0
|     |     |     |     |     |
0 --- 0 --- 1 --- 1 --- 0 --- 0
|     |     |     |     |     |
N --- 1 --- 1 --- 0 --- N --- N
|     |     |     |     |     |
N --- 0 --- 0 --- N --- N --- N
```



Annotation

We benefit from Pulp library in Python for implementation of the project. According to the information we obtained later, there were different solvers in this library. We did out experiments according to the default solver, but it can take some time in cases where the string length is long and the grid size is large. For this, when you convert the solver to CPLEX, it can be done in a shorter time. So, we added both of them to the code with comment and uncomment lines. When you run the code it solves with default solver. If it is necessary, in order to solve them with CPLEX you should do the followings:

For part 1:

- Uncomment line 106,107
- Comment line 103

For part 2:

- Uncomment line 129,130,145
- Comment line 126,146