Optimization TD3

Chensheng LUO,Yue WANG
Track 1, Group 1

March 11th 2022

1 Question 1

Define the total number of objects or boxes as $n = 15$. To translate the two constraints, we can have :

— The box i contains an object and only one :

$$\forall i \in [\![1, n]\!], \sum_{j=1}^{n} x_{i,j} = 1 \tag{1}$$

— The object j is in a box and only one :

$$\forall j \in [\![1, n]\!], \sum_{i=1}^{n} x_{i,j} = 1 \tag{2}$$

To simplify the notation, define the vector $\mathbb{1}_n = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}_{n\times 1}$. With binary matrix $\boldsymbol{x}$, we transform the above equations to :

$$\boldsymbol{x}\mathbb{1}_n = \mathbb{1} \tag{3}$$

$$\mathbb{1}_n^T \boldsymbol{x} = \mathbb{1}_n^T \tag{4}$$

For following simplification, we equally define identity matrix $\mathbb{I}_n = \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & 0 \\ \vdots & \cdots & 1 \end{bmatrix}_{n\times n}$

2 Question 2

2.1 Formulation of question

Let's define a vector $\tilde{\boldsymbol{x}} \in \{0, 1\}^{n^2\times 1}$ as $\tilde{\boldsymbol{x}} = \begin{bmatrix} x_{1,1} & \cdots & x_{n,1} & \cdots & x_{1,n} & \cdots & x_{n,n} \end{bmatrix}^T$.

First, let's express the cost function. Our aim is to minimize the total moving distance. We describe the moving distance vector $\boldsymbol{c} \in \mathbb{R}^{n^2 \times 1}$ as

$$\boldsymbol{c} = \begin{bmatrix} c_{1,1} & \cdots & c_{n,1} & \cdots & c_{i,j} & \cdots & c_{1,n} & \cdots & c_{n,n} \end{bmatrix}^T \tag{5}$$

where

$$\forall (i,j) \in [\![1,n]\!]^2, c_{i,j} = \|O_j - B_i\| \tag{6}$$

and our aim is then to minimize the total distance, *i.e*, $< \boldsymbol{c} | \tilde{\boldsymbol{x}} >$.

Then, let's express the constrains mentioned in section 1

$$L\tilde{\boldsymbol{x}} = \mathbb{1}_{2n} \tag{7}$$

where $L = \begin{bmatrix} L_1 \\ L_2 \end{bmatrix}$, with

$$L_1 = \begin{bmatrix} \mathbb{1}_n^T & 0^{1\times n} & \cdots & 0^{1\times n} \\ 0^{1\times n} & \mathbb{1}_n^T & \cdots & 0^{1\times n} \\ \vdots & \vdots & \ddots & \vdots \\ 0^{1\times n} & 0^{1\times n} & \cdots & \mathbb{1}_n^T \end{bmatrix} \in \mathbb{N}^{n\times n^2} \tag{8}$$

and

$$L_2 = \begin{bmatrix} \mathbb{I}_n & \cdots & \mathbb{I}_n \end{bmatrix} \in \mathbb{N}^{n\times n^2} \tag{9}$$

With above, the total question is then

$$\min_{\tilde{\boldsymbol{x}} \in \{0,1\}^{n^2\times 1}} < \boldsymbol{c} | \tilde{\boldsymbol{x}} > \quad s.t \quad L\tilde{\boldsymbol{x}} = \mathbb{1}_{2n} \tag{10}$$

2.2 Proof of unimodularity

L is total unimodular because

— Its components are 1,-1 or 0.

— $\forall i \in [\![1, n^2]\!]$, $\sum_{j=1}^{2n} L_{j,i} = L_{\lceil \frac{i}{n} \rceil, i} + L_{n+i\%n,i} = 2$.

— $\exists \mathbb{K}_1 = \{1, ..., n\}, \mathbb{K}_2 = \{n+1, ..., 2n\}$ such that $\mathbb{K}_1 \cup \mathbb{K}_2 = [\![1, n^2]\!]$ and $\forall i \in [\![1, n^2]\!], \sum_{j\in\mathbb{K}_1} L_{j,i} = \sum_{j\in\mathbb{K}_2} L_{j,i} = 2$.

2.3 Programming in MATLAB

Following codes are proposed by professor :

```
%% Initialisation
clear all;  close all;  clc
% load the data
T=readtable('PositionsObjects.txt'); PositionsObjects=T.x+1i*T.y;  %
    positions defined by complex numbers
T=readtable('PositionsBoxes.txt');   PositionsBoxes=T.x+1i*T.y;

%% Plot example with object i in the box i (1 in 1, 2 in 2, etc.)
n = length (PositionsObjects);   % number of objects
Boxes = 1:n;
PlotSolution (Boxes, PositionsObjects, PositionsBoxes)
```

First, let's define the vector $\boldsymbol{c}$ and L :

```
% Definition of c
c=zeros(n*n,1);
for i=1:n
    for j=1:n
        c((j-1)*n+i)=norm(PositionsObjects(j)-PositionsBoxes(i));
    end
end

%Definition of L
L1=[];
for i=1:n
    L1=blkdiag(L1,ones(n,1)');
end
L2=[];
for i=1:n
    L2=[L2,eye(n)];
end
L=[L1;L2];
```

Then, let's run the `linprog` for the solution :

```
%Solution
upper_bound=zeros(n*n,1);
lower_bound=ones(n*n,1);
[x_tilde,fval]=linprog(c,[],[],L,ones(2*n,1),upper_bound,lower_bound);
```

Finally, we do the post-treatment of solution, this time we define a function

```
function Boxes=ConvPlot(x_tilde,n)
%Process the post-treatment of solution
x=reshape(x_tilde,n,n);
Boxes=zeros(n,1);
for i=1:n
    Boxes(i)=find(x(:,i));
end
```

and we process

```
%Plot the result
Boxes=ConvPlot(x_tilde,n);
PlotSolution (Boxes, PositionsObjects, PositionsBoxes);
```

It gives the final solution shown as in figure 1.

2.4 A necessary condition for the solution

For a solution, the affectation lines should not have intersections. Consider the following case in figure 2.

Suppose that the solution gives the red affectation lines, then $O_1B_1 + O_2B_2 \geqslant B_1O_2 + B_2O_1$. However, according to the triangular inequality, we have $B_1O_2 + B_2O_1 = B_1C + CO_2 + B_2C + CO_1 > O_1B_1 + O_2B_2$. Therefore, there is a contradiction.

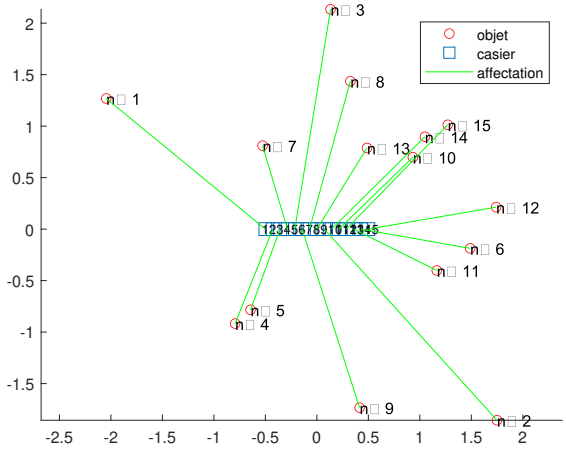

FIGURE 1 – Solution of Question 2

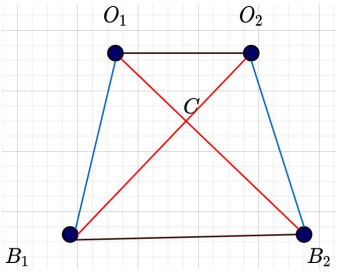

FIGURE 2 – Illustrative example of the necessary condition

3 Question 3

As the object 1 can't be neither in box 1 nor in box n, we should have :

$$x_{1,1} = 0 \tag{11}$$
$$x_{n,1} = 0 \tag{12}$$

So we simply need to modify the upper bound of $x_{1,1}$ and $x_{n,1}$ to 0.

The following code explain this process :

```
%% Q3
upper_bound(1)=0;
upper_bound(n)=0;
[x_tilde,fval]=linprog(c,[],[],L,ones(2*n,1),lower_bound,upper_bound);
Boxes=ConvPlot(x_tilde,n);
PlotSolution (Boxes, PositionsObjects, PositionsBoxes);
```

And we get the final result in figure 3.

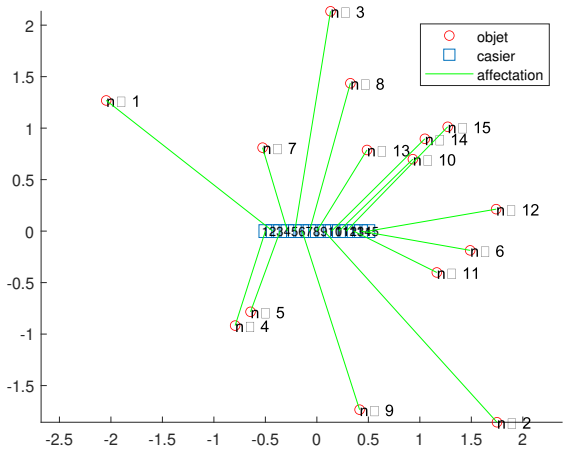

FIGURE 3 – Solution of Question 3

4 Question 4

The condition "object 2 must be in the box located just to the right of the box containing object 3" requires that : if $x_{i,3} = 1$,then $x_{i+1,2} = 1$; if $x_{i,3} = 0$,then $x_{i+1,2} = 0$.

This can be explained as :

$$\forall i \in [\![1, n-1]\!], x_{i+1,2} - x_{i,3} = 0 \tag{13}$$

We then try to explain this with the vector $\tilde{\boldsymbol{x}}$, we have :

$$\left[0^{(n-1)\times(n+1)} \quad \mathbb{I}_{n-1} \quad -\mathbb{I}_{n-1} \quad 0^{(n-1)\times(n^2-3n+1)}\right] \tilde{\boldsymbol{x}} = 0_{(n-1)\times 1} \tag{14}$$

We the explain it to MATLAB code.

Attention !This code just follows the code of section 2.3 and all code of section 3 shouldn't be executed !

```
%Definition of L3
L3=[zeros(n-1,n+1),eye(n-1),-eye(n-1),zeros(n-1,n*n-3*n+1)];
L=[L;L3];
%Solve!
[x_tilde,fval]=linprog(c,[],[],L,[ones(2*n,1);zeros(n-1,1)],lower_bound
    ,upper_bound);
Boxes=ConvPlot(x_tilde,n);
PlotSolution (Boxes, PositionsObjects, PositionsBoxes);
```

And this code gives the solution of figure 4.

5 Question 5

The constraint $\forall i, x_{i,4} + \Sigma_{|k|\leq 3} x_{i+k,5} \leq 1$ means that there should be at least 3 boxes between the box containing object 4 and that containing object 5.

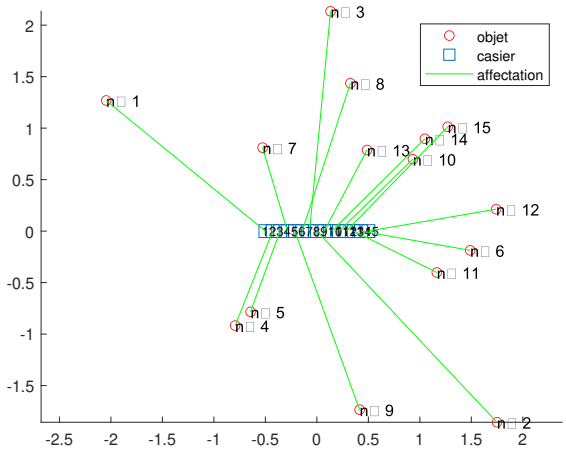

FIGURE 4 – Solution of Question 4

This gives the inequality constraints :

$$A\tilde{x} \leq \mathbb{1}_n \tag{15}$$

where

$$A = \begin{bmatrix} 0^{n\times 3n} & I_n & A_5 & 0^{n\times 10n} \end{bmatrix} \tag{16}$$

$$A_5 = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \cdots 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & \cdots 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & \cdots 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \cdots 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & \cdots 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{17}$$

By converting this into MATLAB code, we have :

Attention !This code just follows the code of section 2.3 and all code of section 3, section 4 shouldn't be executed !

```
%% Q5
%Definition of A5
A5=[];
for i=1:n
    if i<=4
        A5=[A5;ones(1,i+3),zeros(1,n-(i+3))];
    else if i>=12
        A5=[A5;zeros(1,i-4),ones(1,n+4-i)];
        else
            A5=[A5;zeros(1,i-4),ones(1,7),zeros(1,n-i-3)];
```

```
        end
    end
end
A4=eye(15);
A=[zeros(n,3*n),A4,A5,zeros(n,10*n)];
b=ones(n,1);

%Solution
[x_tilde,fval]=intlinprog(c,1:(n*n),A,b,L,ones(2*n,1),lower_bound,
    upper_bound);
Boxes=ConvPlot(x_tilde,n);
PlotSolution (Boxes, PositionsObjects, PositionsBoxes);
```

And this code gives the solution of figure 5.

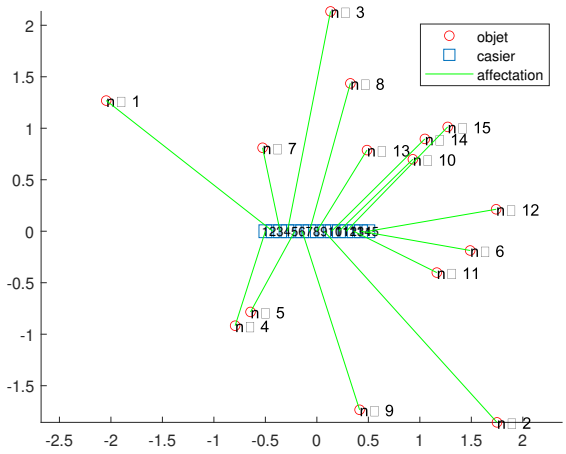

FIGURE 5 – Solution of Question 5

6 Question 6

Similarly, we can use the same formula as in section 5 :

$$\forall i, x_{i,6} - \Sigma_{|k|\leq 3} x_{i+k,7} \leq 0 \tag{18}$$

.

This gives the inequality constraints :

$$A\tilde{x} \leq 0_n \tag{19}$$

where

$$A = \begin{bmatrix} 0^{n\times 5n} & I_n & -A_5 & 0^{n\times 8n} \end{bmatrix} \tag{20}$$

By converting this into MATLAB code, we have :

Attention !This code just follows the code of section 2.3 and all code of section 3, section 4, section 5 shouldn't be executed !

```
%Definition of A
A5=[];
for i=1:n
    if i<=4
        A5=[A5;ones(1,i+3),zeros(1,n-(i+3))];
    else if i>=12
        A5=[A5;zeros(1,i-4),ones(1,n+4-i)];
        else
            A5=[A5;zeros(1,i-4),ones(1,7),zeros(1,n-i-3)];
        end
    end
end
A6=eye(15);
A=[zeros(n,5*n),A6,-A5,zeros(n,8*n)];
b=zeros(n,1);

%Solution
[x_tilde,fval]=intlinprog(c,1:(n*n),A,b,L,ones(2*n,1),lower_bound,
    upper_bound);
Boxes=ConvPlot(x_tilde,n);
PlotSolution (Boxes, PositionsObjects, PositionsBoxes);
```

And this code gives the solution of figure 6.

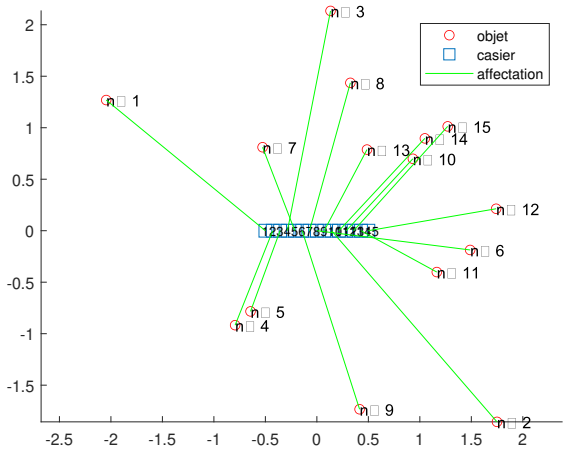

FIGURE 6 – Solution of Question 6

7 Question 7

7.1 Strategy

A simple strategy of verifying the uniqueness of solution is to delete the optimal solution from the feasible domain and try again to solve this question. We see if we can get again the same optimal value.

As we know that there will only be n position in $\tilde{\boldsymbol{x}}$ that the value is 1, other values are all 0. To delete this optimal solution, a simple strategy for this question is to make a scalar product with this solution. If the new optimal solution is the same as the older one, it's not unique. Othervise it's unique.

More precisely, we have the original optimization question $\mathcal{Q}$:

$$\min_{\tilde{\boldsymbol{x}}\in\{0,1\}^{n^2\times 1}} < \boldsymbol{c}|\tilde{\boldsymbol{x}} > \quad s.t \quad A\tilde{\boldsymbol{x}} \leq \boldsymbol{b}, L\tilde{\boldsymbol{x}} = \boldsymbol{d}$$

And it gives an optimal solution $\tilde{\boldsymbol{x}}_1$ with $f_1 =< \boldsymbol{c}|\tilde{\boldsymbol{x}}_1 >$. Then we solve again this question, with additional constrains :

$$\min_{\tilde{\boldsymbol{x}}\in\{0,1\}^{n^2\times 1}} < \boldsymbol{c}|\tilde{\boldsymbol{x}} > \quad s.t \quad \begin{bmatrix} A \\ \tilde{\boldsymbol{x}}_1^T \end{bmatrix} \tilde{\boldsymbol{x}} \leq \begin{bmatrix} \boldsymbol{b} \\ n-1 \end{bmatrix}, L\tilde{\boldsymbol{x}} = \boldsymbol{d}$$

And we see if the solution gives $f_2 = f_1$

7.2 Test result on $\mathcal{P}_2$

We process this strategy on $\mathcal{P}_2$. We remind that the section 4 gives a solution as in figure 4 and the optimal value is 20.5341.Imagine the original optimal solution is `x_tilde1`, we execute :

Attention !This code just follows the code of section 4 and all other codes shouldn't be executed !

```
%New solution
A=x_tilde1';
b=n-1;
[x_tilde2,fval2]=intlinprog(c,1:n^2,A,b,L,[ones(2*n,1);zeros(n-1,1)],
    lower_bound,upper_bound);
%examine
if abs(fval1-fval2)<1e-5
    disp('not unique');
    Boxes=ConvPlot(x_tilde1,n);
    PlotSolution (Boxes, PositionsObjects, PositionsBoxes);

    Boxes=ConvPlot(x_tilde2,n);
    PlotSolution (Boxes, PositionsObjects, PositionsBoxes);
else
    disp('unique');
end
```

And it gives the answer 'unique', with the new optimal solution $20.5343 > 20.5341$.

7.3 Test result on $\mathcal{P}_3$

We process this strategy on $\mathcal{P}_3$. We remind that the section 5 gives a solution as in figure 5 and the optimal value is 20.5816. Imagine the original optimal solution is `x_tilde1`, we execute :

Attention !This code just follows the code of section 5 and all other codes shouldn't be executed !

```
%New solution
A=[A;x_tilde1'];
b=[b;n-1];

[x_tilde2,fval2]=intlinprog(c,1:(n*n),A,b,L,ones(2*n,1),lower_bound,
    upper_bound);
%examine
if abs(fval1-fval2)<1e-5
    disp('not unique');
    Boxes=ConvPlot(x_tilde1,n);
    PlotSolution (Boxes, PositionsObjects, PositionsBoxes);

    Boxes=ConvPlot(x_tilde2,n);
    PlotSolution (Boxes, PositionsObjects, PositionsBoxes);
else
    disp('unique');
end
```

And it gives the new optimal solution 20.5816. So this solution isn't unique. We have another solution proposed as in the figure 7.

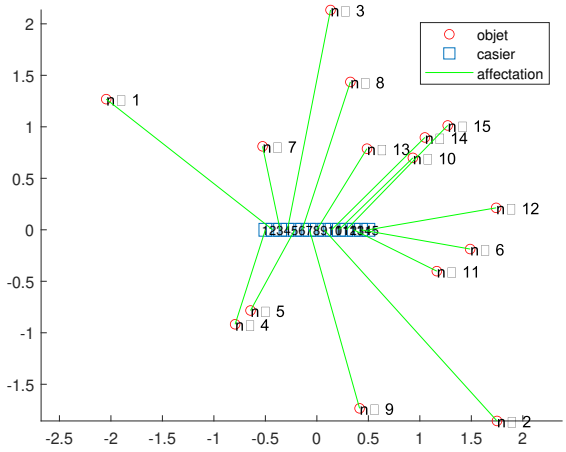

FIGURE 7 – Another Optimal Solution of Question 5

8 Comment on linprog and intlinprog

It can be easily verified that the matrices L in the questions 2,3,4 are TUM. According to the course, the solutions delivered by the simplex to the corresponding LP problems are in $\{0,1\}^{n^2}$. Therefore, the solutions are also the solutions of the corresponding ILP problems. That's why we could use the Matlab function `linprog` instead of `intlinprog`.