



# **Exact Optimization Methods based on Integer Linear Programming**

*Simulação e Otimização*

Mestrado em Engenharia Informática  
Mestrado em Robótica e Sistemas Inteligentes

*Amaro de Sousa, Nuno Lau*

DETI-UA, 2024/2025

# Mathematical programming

- In an ***optimization problem***, the aim is to maximize (or minimize) a given quantity designated as the ***objective*** that depends on a finite number of variables.
- The variables might be independent or might be related between them through one or more ***constraints***.
- A ***mathematical programming problem*** is an optimization problem such that the objective and the constraints are defined by mathematical functions and functional relations.
- A ***mathematical programming model*** describes a mathematical programming problem.

# Mathematical programming model

For a given set of  $n$  variables  $X = \{x_1, x_2, \dots, x_n\}$ , the standard way of defining a Mathematical Programming Model is:

---

Minimize (or Maximize)

$$f(X)$$

Subject to:

$$g_i(X) \leq k_i, \quad i = 1, 2, \dots, m$$

(=)  
( $\geq$ )

---

where:

- $m$  is the number of constraints
- $f(X)$  and all  $g_i(X)$  are mathematical functions of the variables
- $k_i$  are real parameters

## (Mixed Integer) Linear Programming model

- A **Linear Programming (LP)** model is a mathematical programming model where all variables  $X = \{x_1, x_2, \dots, x_n\}$  are non-negative reals and  $f(X)$  and  $g_i(X)$  are linear functions:
  - functions in the form  $a_1 x_1 + a_2 x_2 + \dots + a_n x_n$  where all  $a_i$  are real parameters
- An **Integer Linear Programming (ILP)** model is an LP model where all variables  $X = \{x_1, x_2, \dots, x_n\}$  are non-negative integers.
- A **Mixed Integer Linear Programming (MILP)** model is an LP model where some of the variables  $X = \{x_1, x_2, \dots, x_n\}$  are non-negative integers and others are non-negative reals.

# (Mixed Integer) Linear Programming model

Minimize (or Maximize)

$$c_1 x_1 + c_2 x_2 + \dots + c_n x_n$$

The aim is to assign the values to all variables  $x_1 \dots x_n$  that optimize the objective function

Subject to:

$$a_{11} x_1 + a_{12} x_2 + \dots + a_{1n} x_n \leq k_1$$

$$a_{21} x_1 + a_{22} x_2 + \dots + a_{2n} x_n \leq k_2$$

...

$$a_{m1} x_1 + a_{m2} x_2 + \dots + a_{mn} x_n \leq k_m$$

All constraints must be met by the values assigned to all variables  $x_1 \dots x_n$

- 
- Constraints with ' $\geq$ ' can be formulated with ' $\leq$ ' as:

$$g_i(X) \geq k_i \rightarrow -g_i(X) \leq -k_i$$

- Constraints with '=' can be formulated with ' $\leq$ ' as:

$$g_i(X) = k_i \rightarrow g_i(X) \leq k_i \quad \text{and} \quad -g_i(X) \leq -k_i$$

## Illustrative example

- Consider a logistic operator that has been requested to deliver the following items from its head quarters to a particular destination:

Item $i$ :	1	2	3	4	5	6
Revenue ( $r_i$ ):	2.3	4.5	1.5	5.4	2.9	3.2
Size ( $s_i$ ):	30	75	20	80	35	40

- The company has 2 available vans for this delivery:
  - van 1 has a capacity of 100
  - van 2 has a capacity of 60
- Since  $30+75+20+80+35+40 (=280) > 100+60 (=160)$ , it is not possible to deliver all items with the 2 vans.
- So, the problem is to choose the items to be carried on each van aiming to maximize the total revenue.
- Solving steps:
  - 1<sup>st</sup> – define and implement an ILP model of the optimization problem
  - 2<sup>nd</sup> – solve the ILP model (using an available solver)

## Illustrative example

Item $i$ :	1	2	3	4	5	6
Revenue ( $r_i$ ):	2.3	4.5	1.5	5.4	2.9	3.2
Size ( $s_i$ ):	30	75	20	80	35	40

### VARIABLES DEFINING THE PROBLEM:

means that the value of  $x_1$  can be only 0 or 1

- $x_1$  – Binary variable that, if is 1 in the solution, indicates that item 1 is delivered
- $x_2$  – Binary variable that, if is 1 in the solution, indicates that item 2 is delivered
- ...
- $x_6$  – Binary variable that, if is 1 in the solution, indicates that item 6 is delivered

- $y_{1\_1}$  – Binary variable that, if is 1 in the solution, indicates that item 1 is carried by van 1
- $y_{1\_2}$  – Binary variable that, if is 1 in the solution, indicates that item 1 is carried by van 2

...

- $y_{6\_1}$  – Binary variable that, if is 1 in the solution, indicates that item 6 is carried by van 1
- $y_{6\_2}$  – Binary variable that, if is 1 in the solution, indicates that item 6 is carried by van 2

## Illustrative example

Item $i$ :	1	2	3	4	5	6
Revenue ( $r_i$ ):	2.3	4.5	1.5	5.4	2.9	3.2
Size ( $s_i$ ):	30	75	20	80	35	40

INTEGER LINEAR PROGRAMMING (ILP) MODEL (LP format of 'lp solve' tool):

The objective function is the total revenue of the delivered items

```

Max: + 2.3 x1 + 4.5 x2 + 1.5 x3 + 5.4 x4 + 2.9 x5 + 3.2 x6;
+ 30 y1_1 + 75 y2_1 + 20 y3_1 + 80 y4_1 + 35 y5_1
+ 40 y6_1 <= 100;
+ 30 y1_2 + 75 y2_2 + 20 y3_2 + 80 y4_2 + 35 y5_2
+ 40 y6_2 <= 60;
+ y1_1 + y1_2 = x1;
+ y2_1 + y2_2 = x2;
+ y3_1 + y3_2 = x3;
+ y4_1 + y4_2 = x4;
+ y5_1 + y5_2 = x5;
+ y6_1 + y6_2 = x6;
Bin x1,x2,x3,x4,x5,x6;
Bin y1_1,y1_2,y2_1,y2_2,y3_1,y3_2,y4_1,y4_2,y5_1,y5_2,y6_1,
y6_2;

```

The total size of the items carried on each van must be within the van capacity

If an item is carried in one van, then, the item is delivered

List of binary variables



# Illustrative example: mathematical notation

Parameters:

$n$  – number of items       $r_i$  – revenue of delivering item  $i$ , with  $i = 1, \dots, n$   
 $s_i$  – size of item  $i$ , with  $i = 1, \dots, n$   
 $v$  – number of vans       $c_j$  – capacity of van  $j$ , with  $j = 1, \dots, v$

Variables:

$x_i$  – binary variable that is 1 if item  $i$  is delivered,  $i = 1, \dots, n$   
 $y_{ij}$  – binary variable that is 1 if item  $i$  is carried on van  $j$ ,  $i = 1, \dots, n$  and  $j = 1, \dots, v$

ILP model:      Maximize  $\sum_{i=1}^n r_i x_i$

Subject to:

$$\sum_{i=1}^n s_i y_{ij} \leq c_j \quad , j = 1 \dots v$$

$$\sum_{j=1}^v y_{ij} = x_i \quad , i = 1 \dots n$$

$$x_i \in \{0,1\} \quad , i = 1 \dots n$$

$$y_{ij} \in \{0,1\} \quad , i = 1 \dots n , j = 1, \dots v$$

# Illustrative example – generating LP file with MATLAB

Maximize  $\sum_{i=1}^n r_i x_i$

$\sum_{i=1}^n s_i y_{ij} \leq c_j \quad , j = 1 \dots v$

$\sum_{j=1}^v y_{ij} = x_i \quad , i = 1 \dots n$

$x_i \in \{0,1\} , i = 1 \dots n$

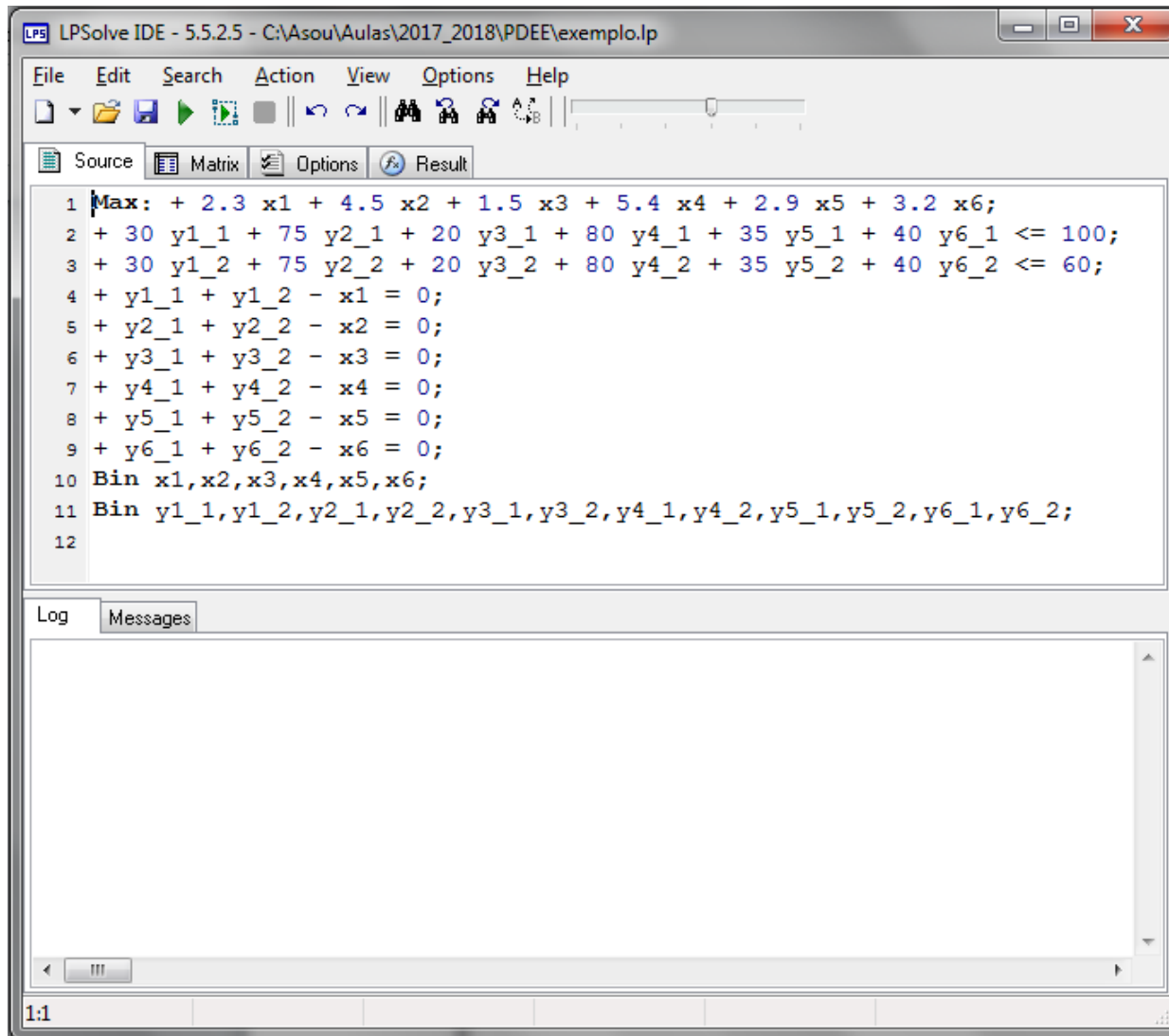
$y_{ij} \in \{0,1\} , i = 1 \dots n , j = 1, \dots v$

```
r= [2.3 4.5 1.5 5.4 2.9 3.2];
s= [30 75 20 80 35 40];
c= [100 60];
n= length(r);
v= length(c);
fid = fopen('example.lp','wt');
fprintf(fid,'max: ');
for i=1:n
    fprintf(fid,'+ %.1f x%d ',r(i),i);
end
fprintf(fid,';\n');
for j=1:v
    for i=1:n
        fprintf(fid,'+ %.1f y%d_%d ',s(i),i,j);
    end
    fprintf(fid,'<= %.1f;\n',c(j));
end
for i=1:n
    for j=1:v
        fprintf(fid,'+ y%d_%d ',i,j);
    end
    fprintf(fid,'= x%d;\n',i);
end
for i=1:n
    fprintf(fid,'Bin x%d;\n',i);
end
for i=1:n
    for j=1:v
        fprintf(fid,'Bin y%d_%d;\n',i,j);
    end
end
fclose(fid);
```

# ILP solvers

- There are various commercial software packages providing algorithms to solve LP and ILP problems such as: CPLEX, Gurobi, XPRESS, etc...
- The 'lpsolve' IDE is a free software.
- Download 'lpsolve':  
<http://sourceforge.net/projects/lpsolve/>
- Help:  
<http://lpsolve.sourceforge.net/5.5/>

# Illustrative example – using ‘lpsolve’ (1)

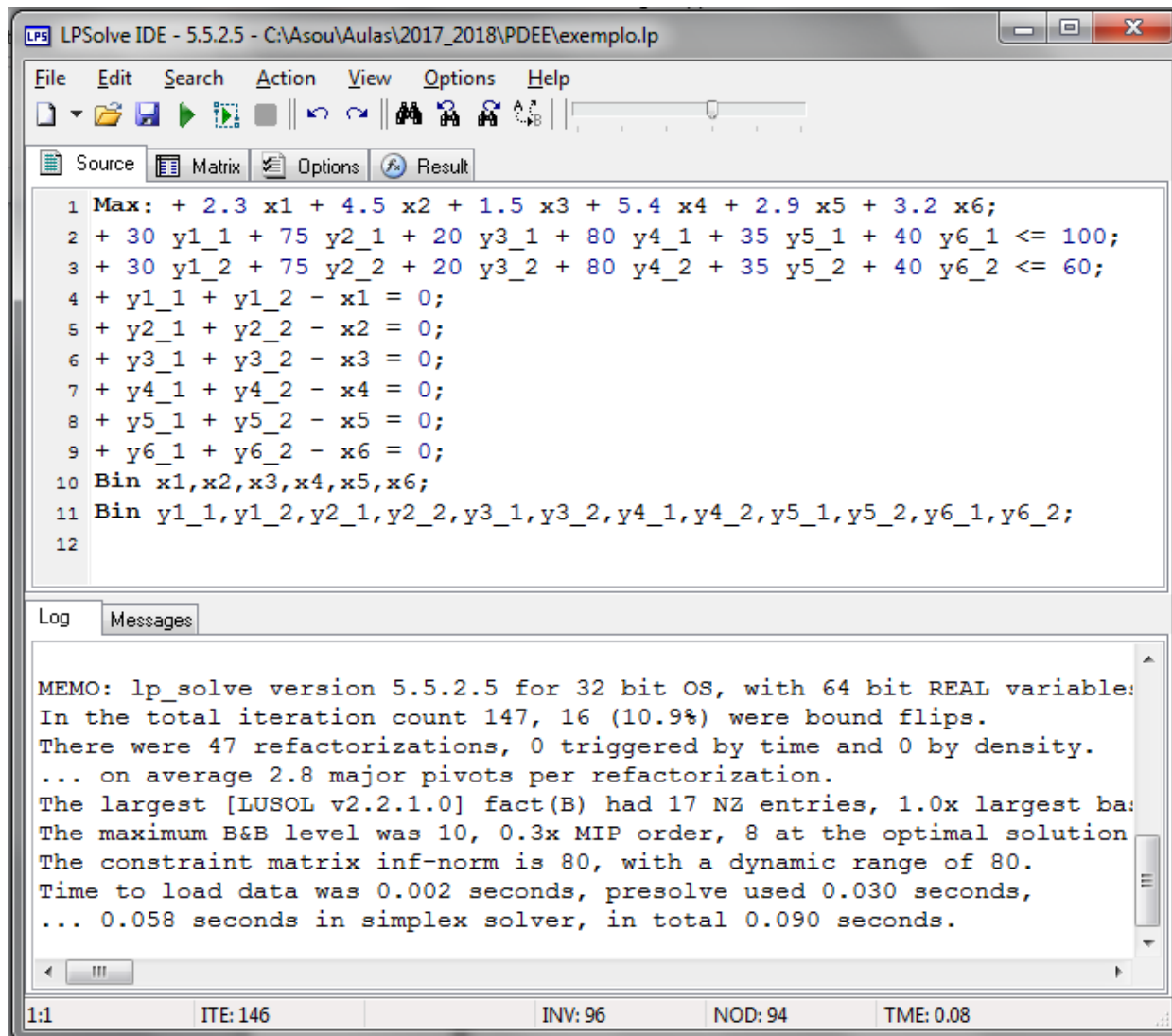


The screenshot shows the LPSolve IDE interface. The title bar reads "LPSolve IDE - 5.5.2.5 - C:\Asou\Aulas\2017\_2018\PDEE\exemplo.lp". The menu bar includes File, Edit, Search, Action, View, Options, and Help. The toolbar contains icons for file operations and solving. The "Source" tab is active, displaying the following code:

```
1 Max: + 2.3 x1 + 4.5 x2 + 1.5 x3 + 5.4 x4 + 2.9 x5 + 3.2 x6;  
2 + 30 y1_1 + 75 y2_1 + 20 y3_1 + 80 y4_1 + 35 y5_1 + 40 y6_1 <= 100;  
3 + 30 y1_2 + 75 y2_2 + 20 y3_2 + 80 y4_2 + 35 y5_2 + 40 y6_2 <= 60;  
4 + y1_1 + y1_2 - x1 = 0;  
5 + y2_1 + y2_2 - x2 = 0;  
6 + y3_1 + y3_2 - x3 = 0;  
7 + y4_1 + y4_2 - x4 = 0;  
8 + y5_1 + y5_2 - x5 = 0;  
9 + y6_1 + y6_2 - x6 = 0;  
10 Bin x1,x2,x3,x4,x5,x6;  
11 Bin y1_1,y1_2,y2_1,y2_2,y3_1,y3_2,y4_1,y4_2,y5_1,y5_2,y6_1,y6_2;  
12
```

Below the code editor is a "Log" and "Messages" section, which is currently empty. The status bar at the bottom shows "1:1".

## Illustrative example – using 'lpsolve' (2)



The screenshot displays the LPSolve IDE interface. The title bar indicates the file path: C:\Asou\Aulas\2017\_2018\PDEE\exemplo.lp. The menu bar includes File, Edit, Search, Action, View, Options, and Help. The toolbar contains icons for file operations and solver controls. The 'Source' tab is active, showing the following LP model:

```
1 Max: + 2.3 x1 + 4.5 x2 + 1.5 x3 + 5.4 x4 + 2.9 x5 + 3.2 x6;  
2 + 30 y1_1 + 75 y2_1 + 20 y3_1 + 80 y4_1 + 35 y5_1 + 40 y6_1 <= 100;  
3 + 30 y1_2 + 75 y2_2 + 20 y3_2 + 80 y4_2 + 35 y5_2 + 40 y6_2 <= 60;  
4 + y1_1 + y1_2 - x1 = 0;  
5 + y2_1 + y2_2 - x2 = 0;  
6 + y3_1 + y3_2 - x3 = 0;  
7 + y4_1 + y4_2 - x4 = 0;  
8 + y5_1 + y5_2 - x5 = 0;  
9 + y6_1 + y6_2 - x6 = 0;  
10 Bin x1,x2,x3,x4,x5,x6;  
11 Bin y1_1,y1_2,y2_1,y2_2,y3_1,y3_2,y4_1,y4_2,y5_1,y5_2,y6_1,y6_2;  
12
```

The 'Log' tab is also visible, displaying the following message:

```
MEMO: lp_solve version 5.5.2.5 for 32 bit OS, with 64 bit REAL variables:  
In the total iteration count 147, 16 (10.9%) were bound flips.  
There were 47 refactorizations, 0 triggered by time and 0 by density.  
... on average 2.8 major pivots per refactorization.  
The largest [LUSOL v2.2.1.0] fact(B) had 17 NZ entries, 1.0x largest bas  
The maximum B&B level was 10, 0.3x MIP order, 8 at the optimal solution  
The constraint matrix inf-norm is 80, with a dynamic range of 80.  
Time to load data was 0.002 seconds, presolve used 0.030 seconds,  
... 0.058 seconds in simplex solver, in total 0.090 seconds.
```

The status bar at the bottom shows the following information: 1:1, ITE: 146, INV: 96, NOD: 94, TME: 0.08.

## Illustrative example – using ‘Ipsolve’ (3)

The screenshot displays the LPSolve IDE interface. The 'Result' tab is active, showing the optimal solution for the problem. The 'Variables' table is as follows:

Variables	MILP ...	MILP ...	result
	9.2	10.1	10.1
x1	1	0	0
x2	0	0	0
x3	1	1	1
x4	1	1	1
x5	0	0	0
x6	0	1	1
y1_1	0	0	0
y2_1	0	0	0
y3_1	1	1	1
y4_1	1	1	1

Annotations on the table:

- A red arrow points to the 'result' column header, with the text "Total revenue is 10.1".
- A blue arrow points to the 'result' column, with the text "Items 3, 4 and 6 are selected to be delivered".

The 'Log' tab shows the following message:

```
MEMO: lp_solve version 5.5.2.5 for 32 bit OS, with 64 bit REAL variable:  
In the total iteration count 147, 16 (10.9%) were bound flips.  
There were 47 refactorizations, 0 triggered by time and 0 by density.  
... on average 2.8 major pivots per refactorization.  
The largest [LUSOL v2.2.1.0] fact(B) had 17 NZ entries, 1.0x largest base  
The maximum B&B level was 10, 0.3x MIP order, 8 at the optimal solution  
The constraint matrix inf-norm is 80, with a dynamic range of 80.  
Time to load data was 0.002 seconds, presolve used 0.030 seconds,  
... 0.058 seconds in simplex solver, in total 0.090 seconds.
```

At the bottom of the window, the status bar shows: 1:1 ITE: 146 INV: 96 NOD: 94 TME: 0.08

Problem  
solved  
in 0.09  
seconds

## Illustrative example: using 'Ipsolve' (4)

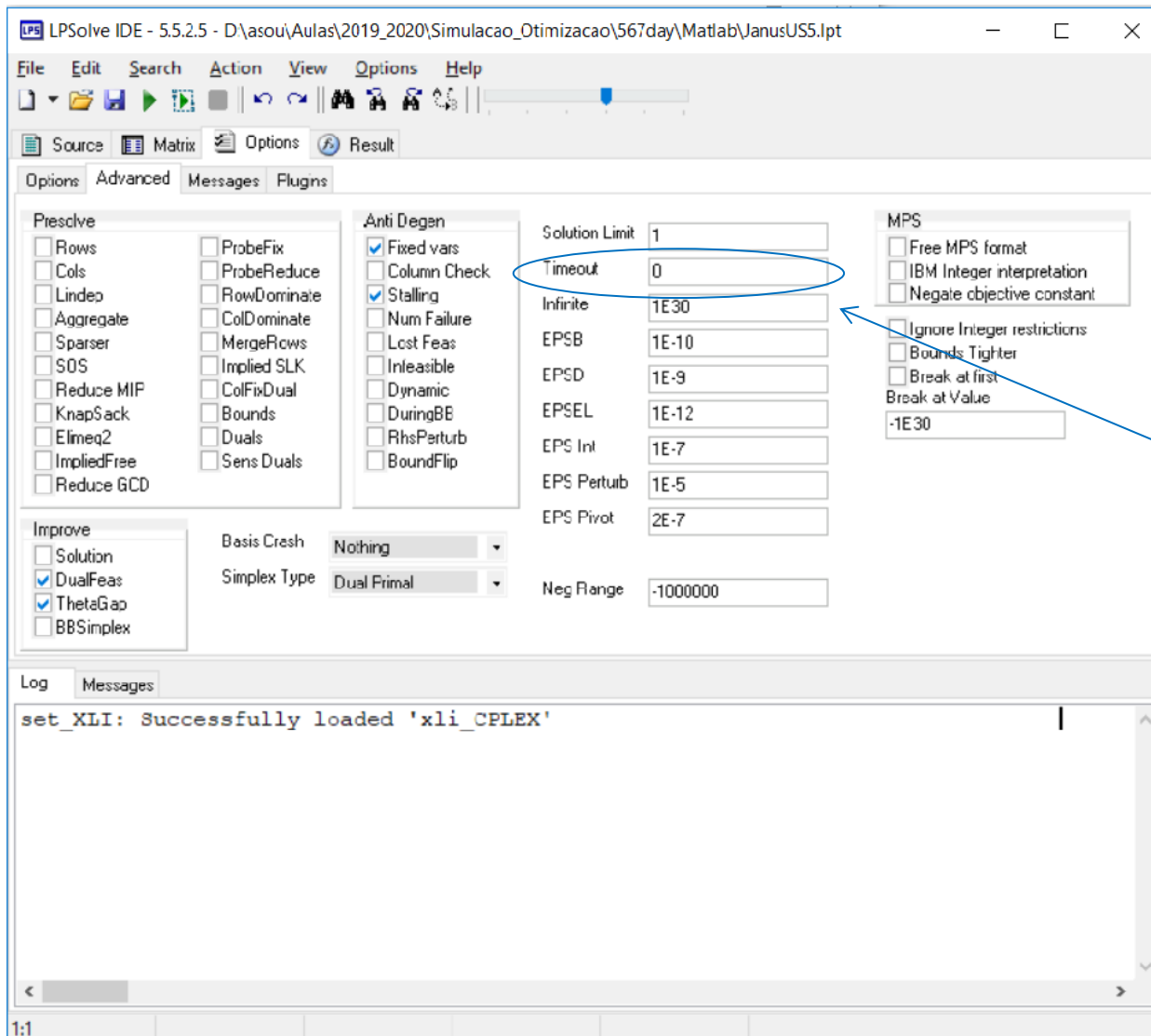
By default, 'Ipsolve' runs until it finds an optimal solution

In hard problems, the running time can be too long to reach an optimal solution

You can set a Timeout (in seconds)

If Timeout is reached, 'Ipsolve' provides:

- the best solution found
- a percentage gap of the maximum difference between the best value and the optimal value.



## Practical Assignment 9

- Use the provided MATLAB script to generate a file named *example.lp* with the ILP description of illustrative example.
  - Use 'lpsolve' to solve the illustrative example.
  - Register the optimal solution and the runtime taken by 'lpsolve' to obtain it.
- Change the MATLAB script to generate a new LP file of the problem for the items and vans defined below.
  - Use 'lpsolve' to solve this new problem.
  - Register the optimal solution and the runtime taken by 'lpsolve' to obtain it. Compare this runtime with the runtime while solving the previous optimization problem.

Items:

$i$	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$r_i$	2.3	4.5	1.5	5.4	2.9	3.2	5.9	2.2	5.4	1.4	2.3	2.1	2.7	3.8
$s_i$	30	75	20	80	35	40	85	15	70	20	25	15	40	50

Vans:

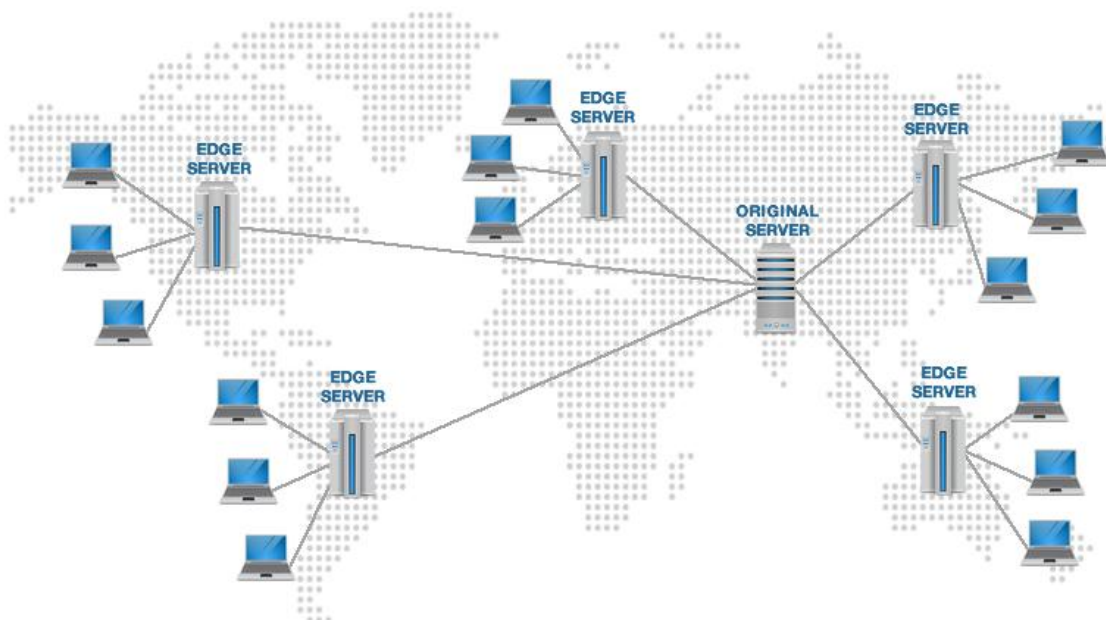
$j$	1	2	3	4	5
$c_j$	100	100	60	60	60



# Server Node Selection Problems

## Motivation:

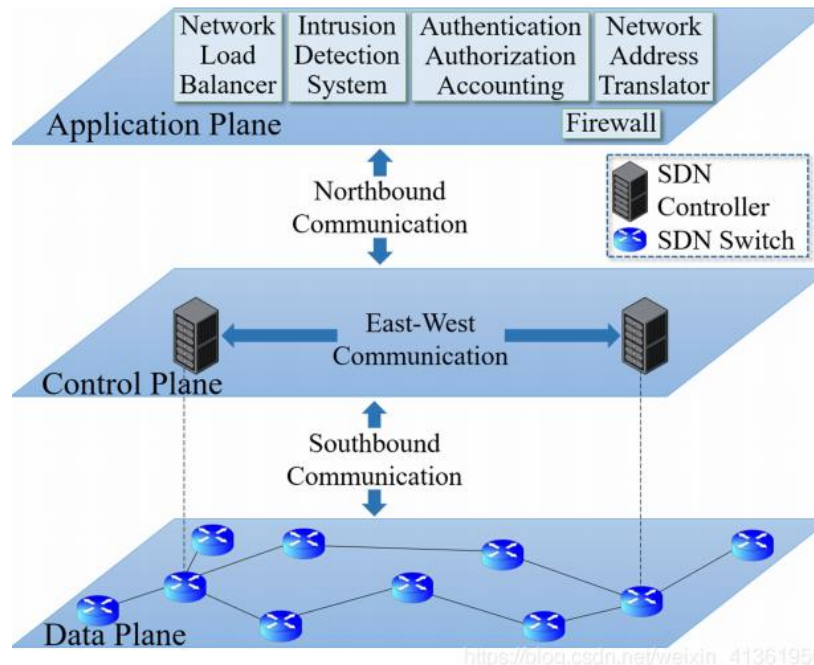
- In Content Delivery Networks (CDNs), content is replicated over Data Centers (DCs) on different locations to improve delivery capacity and reduce delivery delay: clients are served by their closest DC through the shortest path.



# Server Node Selection Problems

## Motivation:

- In Software Defined Networks (SDNs), control plane is centralized in a few controllers (nearly) collocated with switches: switches communicate with their closest controller through the shortest path to establish routing tables.



# Server Node Selection Problems

## Motivation:

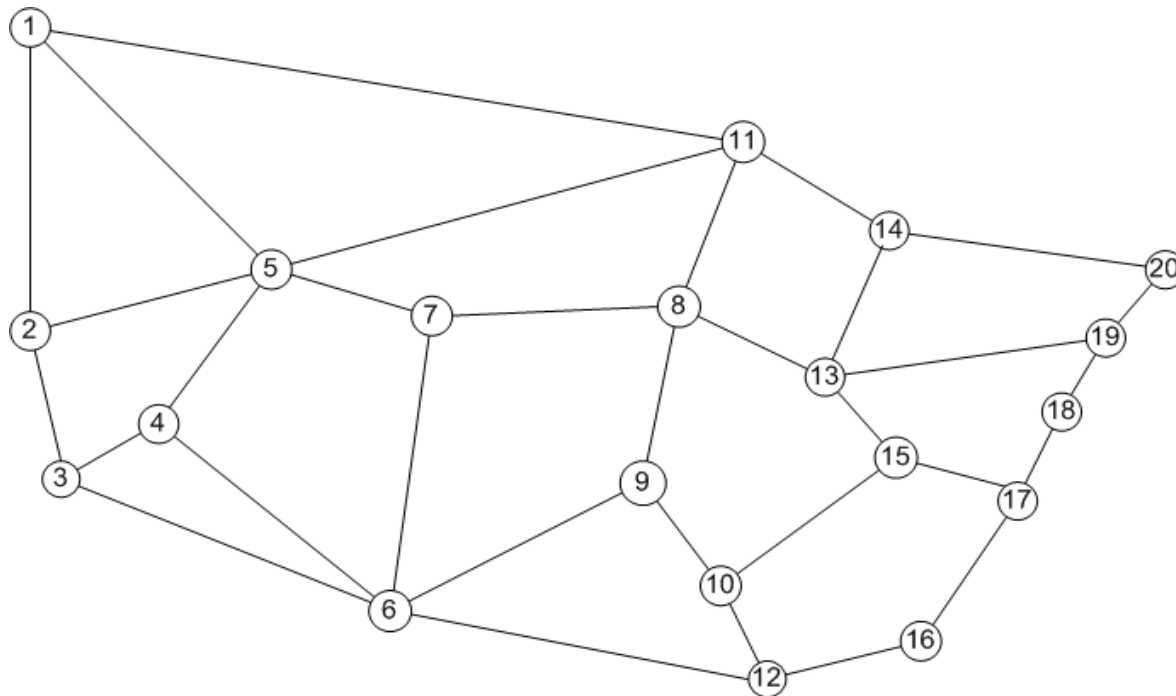
In both cases (CDNs and SDNs):

- some nodes must be selected to host servers (named server nodes):
  - DCs to host content replicas in CDNs
  - switches to connect a controller in SDNs;
- nodes (content clients in CDNs / switches in SDNs) communicate with their closest server through the shortest path provided by the network.

Server node selection problems aim to identify the best server nodes on a given network.

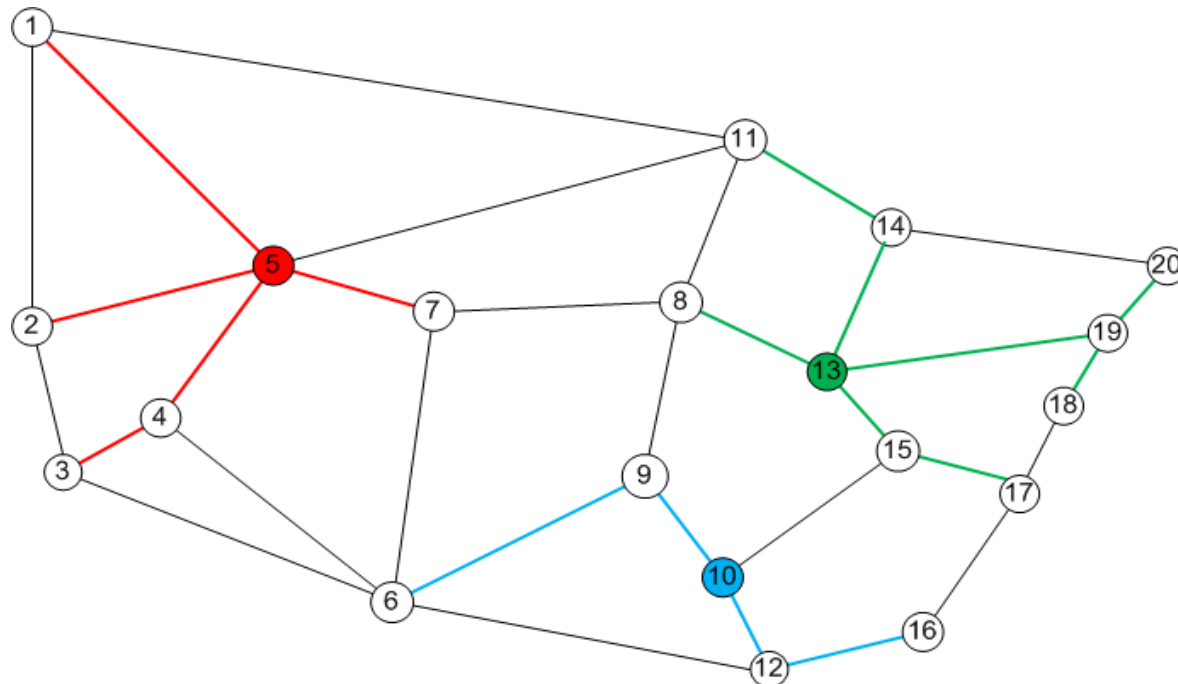
# Server Node Selection Problems

Network example:



# Server Node Selection Problems

- In the 3 selected server nodes, each node communicates with the closest server through the shortest path:

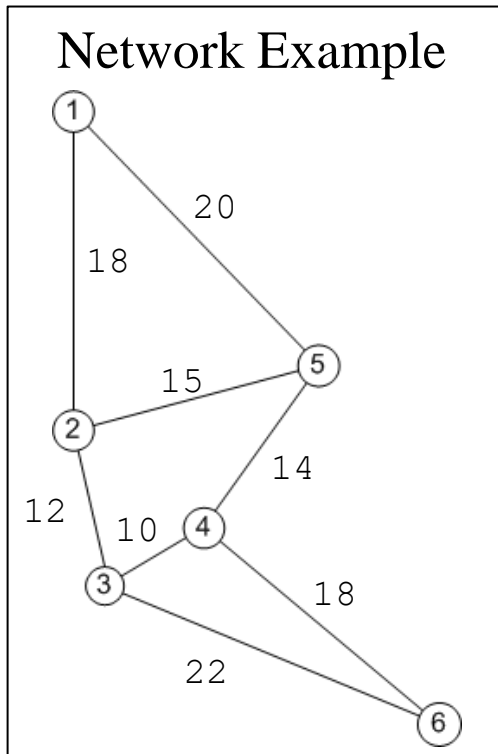


- Note that whatever server nodes, the length of the shortest path between any two nodes is always the same. 21

# Determining shortest path lengths in MATLAB

- $L$  is a square matrix where  $L(i, j)$  is the length of arc  $(i, j)$
- $D$  is a square matrix where  $D(i, j)$  is the shortest path length from node  $i$  to node  $j$

```
L= [ 0 18 0 0 20 0  
    18 0 12 0 15 0  
      0 12 0 10 0 22  
      0 0 10 0 14 18  
    20 15 0 14 0 0  
      0 0 22 18 0 0];
```



```
>> G= graph(L);  
>> D= distances(G)
```

D =

0	18	30	34	20	52
18	0	12	22	15	34
30	12	0	10	24	22
34	22	10	0	14	18
20	15	24	14	0	32
52	34	22	18	32	0

# PROBLEM 1:

## Server Node Selection Problem – Minimizing Delay

- Consider a network modelled by a graph  $G = (N, A)$  such that each arc  $(i, j) \in A$  has an associated length  $l_{ij}$  (it represents the delay of the arc).
- For a given number  $n$  of servers, the aim is to select  $n$  server nodes (i.e., nodes to locate servers) minimizing the average shortest path length from each node to its closest server.
- Parameters:  
 $\delta_s^i$  – precomputed delay of the shortest path from node  $s \in N$  to node  $i \in N$  (for  $s = i$ ,  $\delta_i^i = 0$ ).
- Variables:  
 $z_i$  – binary variable that is 1 if node  $i \in N$  is selected as a server node; 0 otherwise.  
 $g_s^i$  – binary variable that is 1 if node  $s \in N$  is served by node  $i \in N$ ; 0 otherwise.

# PROBLEM 1:

## Server Node Selection Problem – Minimizing Delay

- ILP model:

Minimize  $\sum_{s \in N} \sum_{i \in N} (\delta_s^i g_s^i)$  ← Sum of the delays of all paths  
(minimizing the average or the sum provides  
the same optimal solution as one value is  
the other value divided by  $|N|$ )

- Subject to:

$\sum_{i \in N} z_i = n$  ←  $n$  server nodes must be selected

$\sum_{i \in N} g_s^i = 1$  ,  $s \in N$  ← One server must be assigned to each node

$g_s^i \leq z_i$  ,  $s \in N, i \in N$  ← The server assigned to each node  
must be a server node

$z_i \in \{0,1\}$  ,  $i \in N$

$g_s^i \in \{0,1\}$  ,  $s \in N, i \in N$



# PROBLEM 1:

## Server Node Selection Problem – Minimizing Delay

### Problem Variant 1:

- In CDNs, each node  $s \in N$  might have an associated number of clients  $u_s$ .
- For a given number  $n$  of servers, the aim is to select  $n$  server nodes minimizing the average shortest path length from each client (instead of node) to its closest server.
- A valid ILP is given by the basic model of PROBLEM 1 changing the objective function to the following weighted sum (weights given by  $u_s$  values):

$$\text{Minimize } \sum_{s \in N} \left( u_s \sum_{i \in N} (\delta_s^i g_s^i) \right)$$

# PROBLEM 1:

## Server Node Selection Problem – Minimizing Delay

### Problem Variant 2:

- In wide geographical SDNs, the delay between controllers must be bounded by a given maximum value.
- For a given number  $n$  of servers, the aim is to select  $n$  server nodes:
  - minimizing the average shortest path length from each node to its closest server
  - guaranteeing that the shortest path length between any pair of servers is not higher than a given parameter  $C_{max}$ .
- A valid ILP is given by the basic model of PROBLEM 1 adding the following constraints:

$$z_i + z_j \leq 1 \quad , i \in N, j \in N: \delta_i^j > C_{max}$$

## PROBLEM 2:

### Server Node Selection Problem – Minimizing Cost

Motivation: in CDNs, content hosting has an associated cost; the CDN operator aims to minimize replica location costs guaranteeing a maximum delay from clients to closest content replica

- Consider a network modelled by a graph  $G = (N, A)$  such that each arc  $(i, j) \in A$  has an associated length  $l_{ij}$  (representing the delay) and each node  $i \in N$  has an associated cost  $c_i$  to be used as a server node.
- The aim is to select a set of server nodes:
  - minimizing the cost of the server nodes
  - guaranteeing that the shortest path length from each node to its closest server is not higher than  $L_{max}$

# PROBLEMS 1 and 2: Server Node Selection Problem

## Minimizing Delay:

Minimize  $\sum_{s \in N} \sum_{i \in N} (\delta_s^i g_s^i)$

Subject to:

$$\sum_{i \in N} z_i = n$$

## Minimizing Cost:

Minimize  $\sum_{i \in N} (c_i z_i)$

Subject to:

$$\sum_{i \in N} (\delta_s^i g_s^i) \leq L_{max} \quad , s \in N$$

$$\sum_{i \in N} g_s^i = 1 \quad , s \in N$$

$$g_s^i \leq z_i \quad , s \in N, i \in N$$

$$z_i \in \{0,1\} \quad , i \in N$$

$$g_s^i \in \{0,1\} \quad , s \in N, i \in N$$

## Practical Assignment 10

- Consider the graph  $G = (N, E)$  of Net300 network topology (files: `Nodes300.txt`, `Links300.txt` and `L300.txt`).
- Consider the server node selection problem of selecting a set of  $n$  server nodes that minimize the shortest path length from each node to its closest server.
- Develop a MATLAB script to generate a ILP description of the optimization problem in LP format.
- Solve the problem with 'lpsolve' for  $n = 12$  server nodes. Compare these results with the ones obtained by metaheuristics in Practical Assignments 7 and 8.