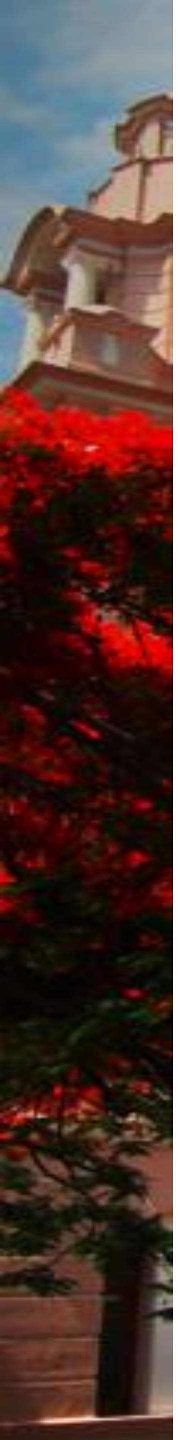
MSBA7003 Quantitative Analysis Methods

07 Mixed Integer Programming

Agenda

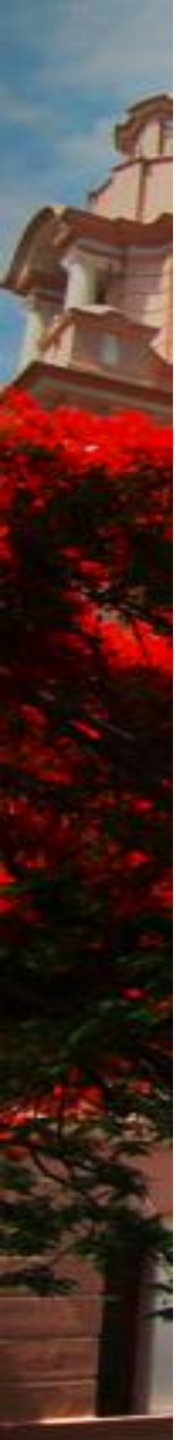
- The Use of Binary Variables
- Applications of Mixed Integer Programming
- Large Scale Problems
- Linear Programming Relaxations



Modeling with 0-1 Variables

- 0-1 (binary) variables are very useful in practical problems.
  - Making a selection among a set of choices
  - Discrete (either-or) choices with fixed costs
  - Dependent selections

- An LP with 0-1 variables is called a Mixed Integer Program.

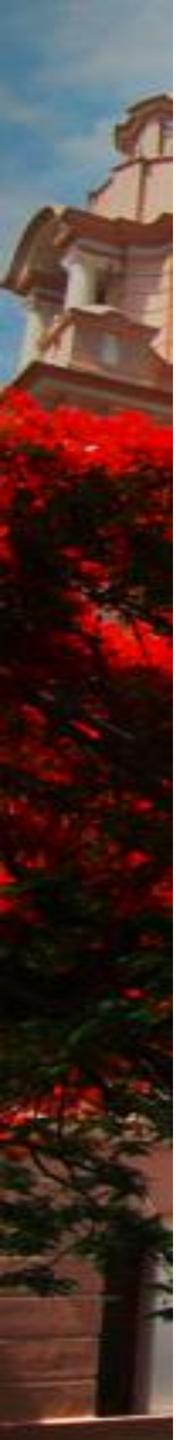


Modeling with 0-1 Variables

- **Making a selection**
- Quemo Chemical Company is considering three possible improvement projects:

Project	NPV	Required Investment	
		Year 1	Year 2
(1) Catalytic Converter	\$25,000	\$8,000	\$7,000
(2) Software Upgrade	\$18,000	\$6,000	\$4,000
(3) Warehouse Expansion	\$32,000	\$12,000	\$8,000
	Available Funds	\$20,000	\$16,000

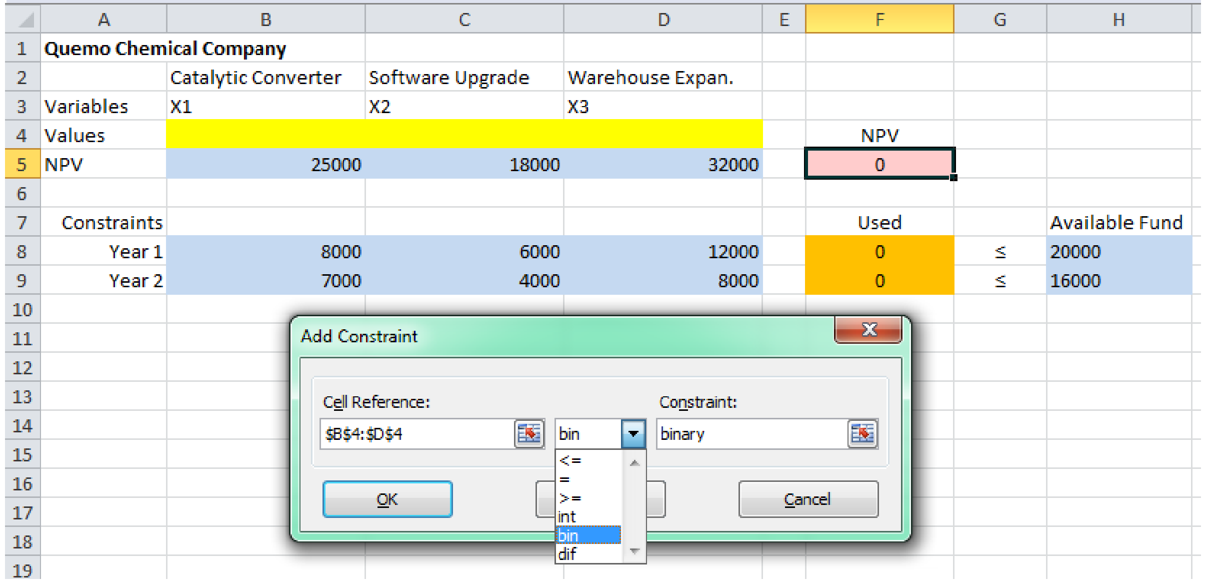
- Which project(s) to undertake to maximize NPV?



Modeling with 0-1 Variables

- Define the decision variables as

$$X_i = \begin{cases} 1 \text{ if project } (i) \text{ is funded} \\ 0 \text{ otherwise} \end{cases}$$



- The mathematical statement of the problem:

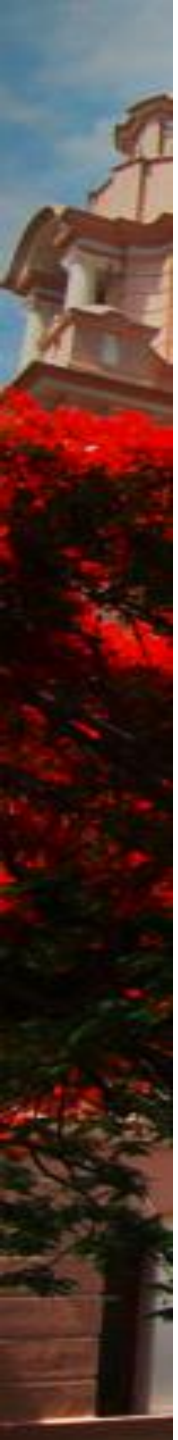
$$\text{Max NPV} = 25{,}000X_1 + 18{,}000X_2 + 32{,}000X_3$$

Subject to

$$8{,}000X_1 + 6{,}000X_2 + 12{,}000X_3 \leq 20{,}000$$

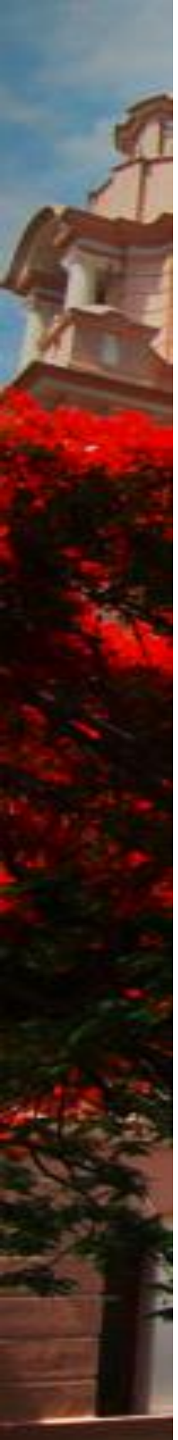
$$7{,}000X_1 + 4{,}000X_2 + \;\; 8{,}000X_3 \leq 16{,}000$$

$$X_1, X_2, X_3 \in \{0,1\}$$



Modeling with 0-1 Variables

- **Modeling dependent selections**
- Suppose that the catalytic converter could be purchased only if the software was upgraded.
- Add a constraint: $X_1 \leq X_2$.
- What if the two projects must be undertaken together?
- Add the constraint: $X_1 = X_2$.



Modeling with 0-1 Variables

- **Modeling fixed and variable costs**

- Suppose there is a fourth option: a marketing program to build the brand name. To start the program, the company has to hire a marketing team, which costs $10,000 in year 1. The company can then decide the amount of money to invest in year 2. The NPV should be 1.5 times the year-2 investment minus the hiring cost.

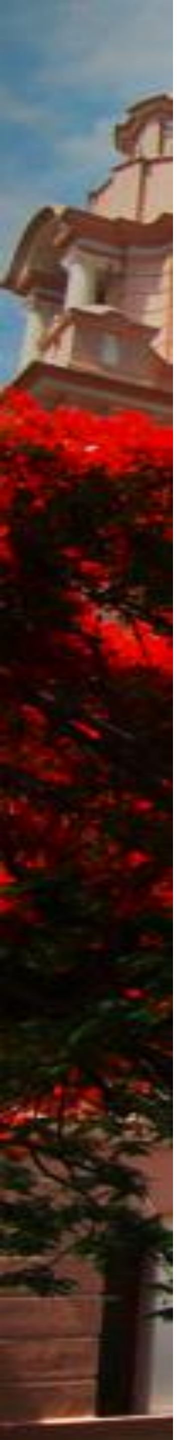
- Add variables X_4 and M_4.

Modeling with 0-1 Variables

- The new problem can be modeled as follows:

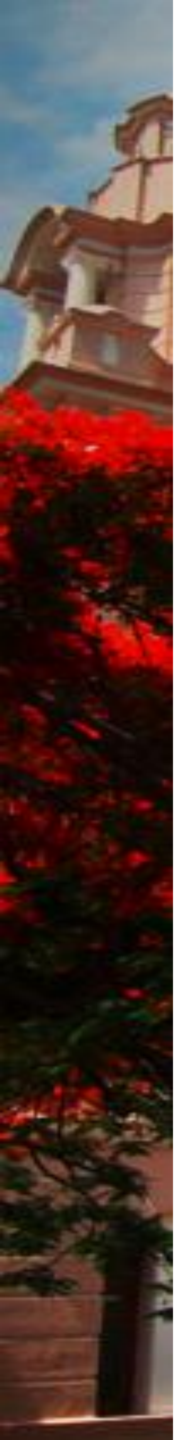
- Max $\text{NPV} = 25{,}000X_1 + 18{,}000X_2 + 32{,}000X_3 + 1.5M_4 - 10{,}000X_4$
- New constraints:
  - $8{,}000X_1 + 6{,}000X_2 + 12{,}000X_3 + 10{,}000X_4 \leq 20{,}000$
  - $7{,}000X_1 + 4{,}000X_2 + \ 8{,}000X_3 \ + \qquad M_4 \leq 16{,}000$
  - $M_4 \leq 16{,}000X_4$
  - $M_4 \geq 0$ and $X_1, X_2, X_3, X_4 \in \{0,1\}$

This constraint is to ensure $M_4 = 0$ when $X_4 = 0$. The control limit can be any positive number greater than 16,000.



In-class Exercises

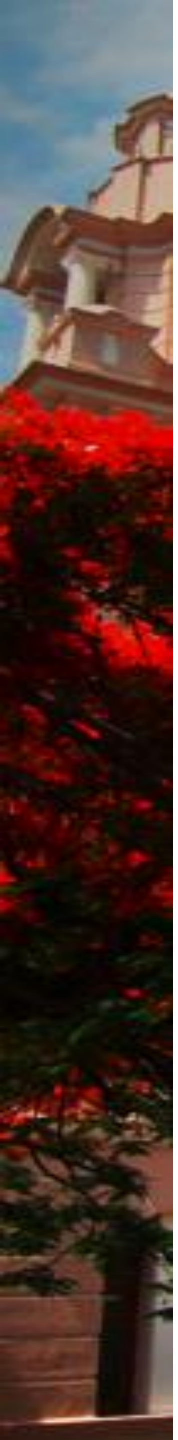
- How to model "at most one can be selected between catalytic converter and warehouse expansion?"
- How to model "catalytic converter could be purchased only when either the software upgrading or the marketing program was undertaken but not both?"
- How to formulate the model if, for the marketing program, an investment could be made in year 1 in addition to the $10,000 needed to hire marketing team. The NPV is 1.5 times the total marketing investment in two years, minus the hiring cost.



Truck Loading Problem

- Goodman Shipping Co. is deciding which items to load on a truck so as to maximize the total value shipped.
- The truck has a capacity of 10,000 pounds and the following items are awaiting shipment.

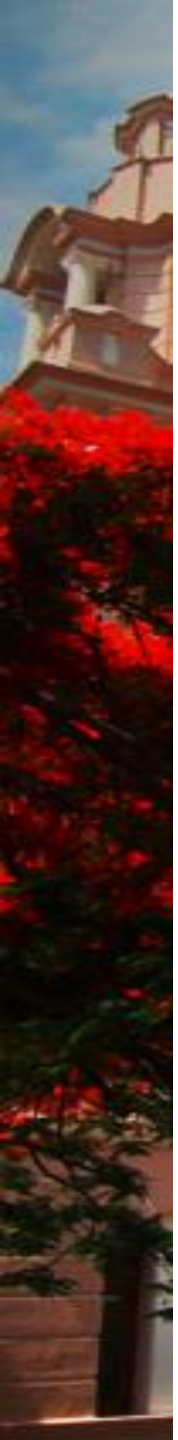
ITEM	VALUE ($)	WEIGHT (lbs)
1	22,500	7,500
2	24,000	7,500
3	8,000	3,000
4	9,500	3,500
5	11,500	4,000
6	9,750	3,500



Truck Loading Problem

- All the decision variables are binary.

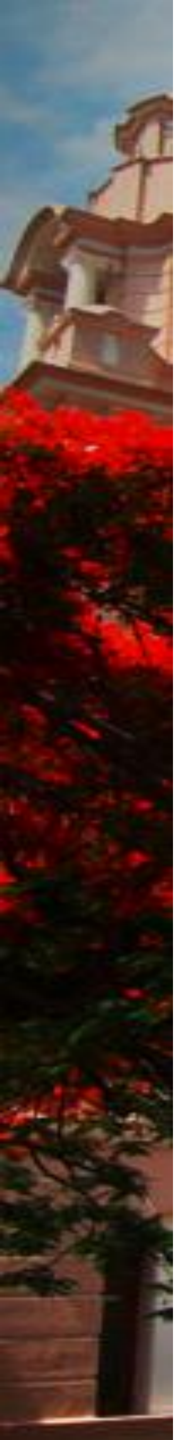
	A	B	C	D	E	F	G	H	I	J	K
1	**Goodman Shipping Co.**										
2											
3	Item	1	2	3	4	5	6				
4	Variable	X1	X2	X3	X4	X5	X6				
5	Decision	0	0	1	1	0	1		Total Value		
6	Value	22500	24000	8000	9500	11500	9750		27250		
7									Total Weight		Capacity
8	Weight	7500	7500	3000	3500	4000	3500		10000	≤	10000



Truck Loading (Assignment Problem)

- If Goodman Shipping Co. must ship all the items and they can use more than one truck in their fleet. What is the loading plan that minimizes the unused capacity.

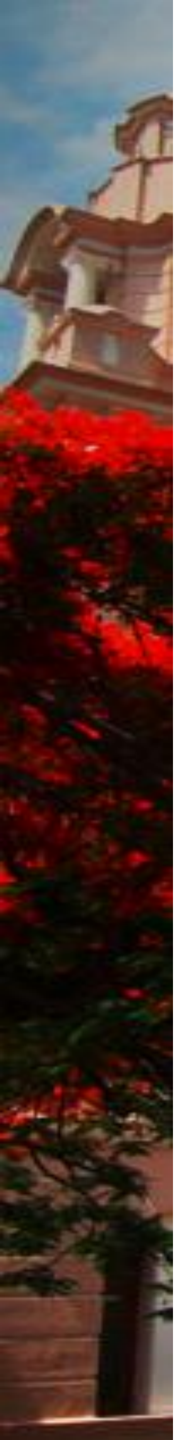
Truck No.	1	2	3	4	5	6
Capacity (lbs)	10,000	5,000	12,000	8,000	4,500	4,000



Truck Loading (Assignment Problem)

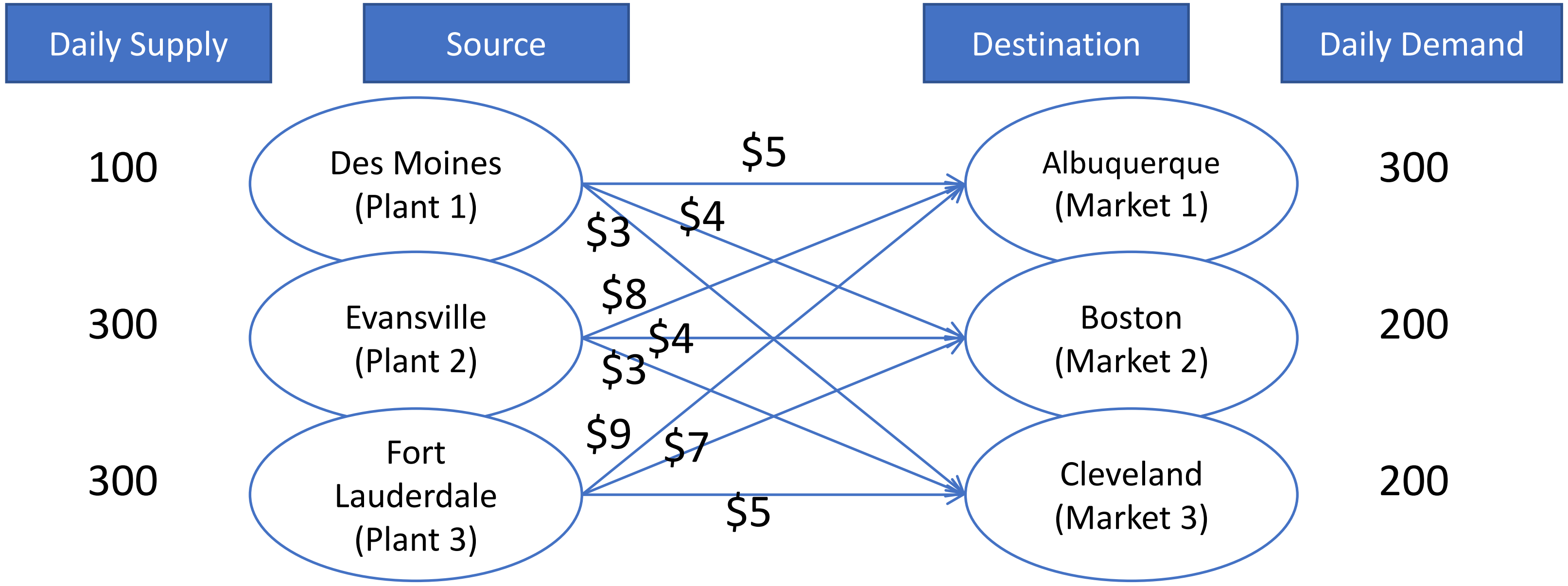
- All the decision variables are binary.

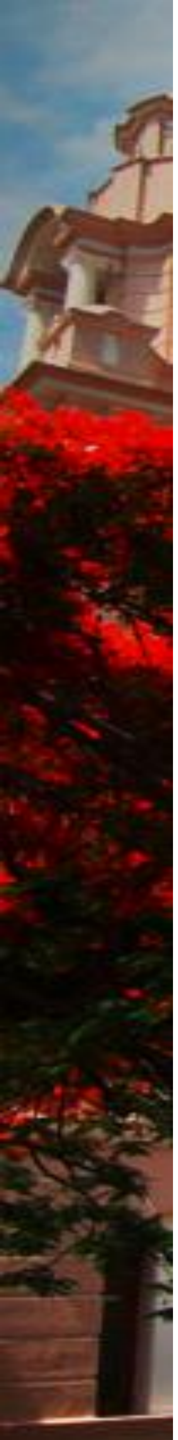
Goodman Shipping Co.												
Item	1	2	3	4	5	6	Total Wasted Capacity					
Weight	7500	7500	3000	3500	4000	3500	1000				Available	Spare
Truck							Capacity	Used?	Load		Capacity	Capacity
1	0	0	1	1	0	1	10000	1	10000	≤	10000	0
2	0	0	0	0	0	0	5000	0	0	≤	0	0
3	1	0	0	0	1	0	12000	1	11500	≤	12000	500
4	0	1	0	0	0	0	8000	1	7500	≤	8000	500
5	0	0	0	0	0	0	4500	0	0	≤	0	0
6	0	0	0	0	0	0	4000	0	0	≤	0	0
Shipped?	1	1	1	1	1	1						
	=	=	=	=	=	=						
	1	1	1	1	1	1						



Supply Chain Planning

- The Executive Furniture Corporation is faced with the following supply chain planning problem. There are three possible locations to operate a plant, and there are three possible markets.





Supply Chain Planning

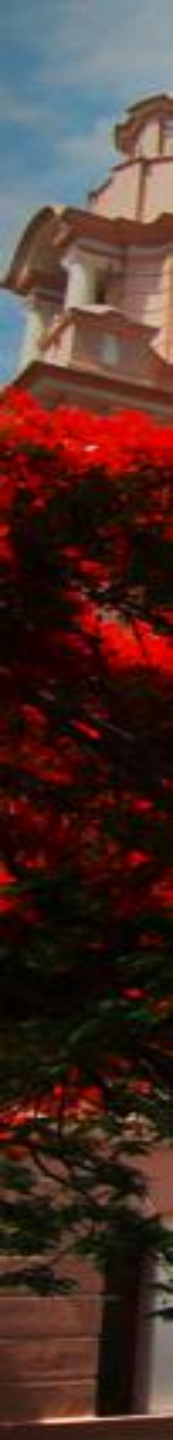
- Suppose there is a daily fixed cost of running each plant and the selling prices at the three destinations differ.
- Which plants and destinations should be chosen to maximize profit?

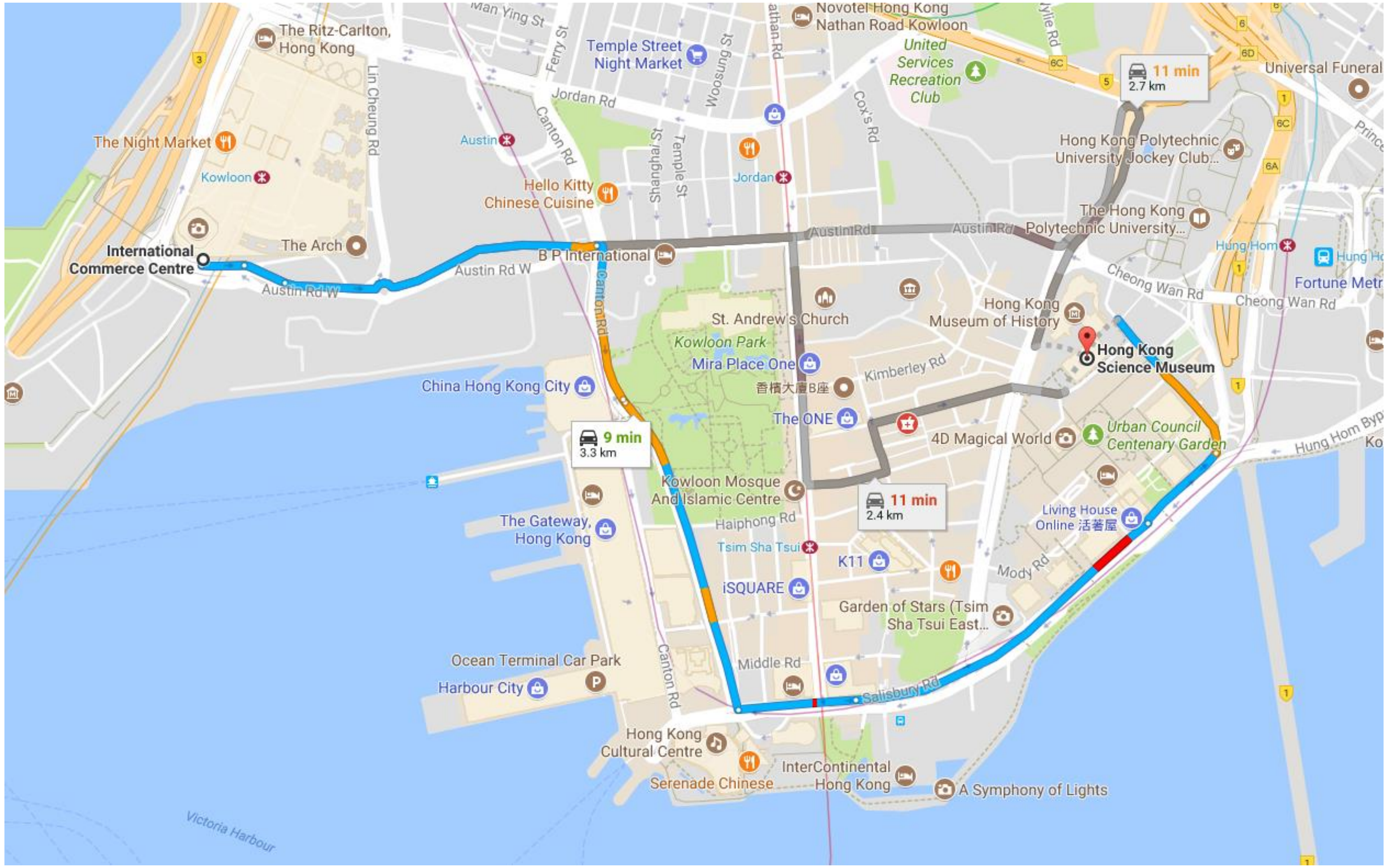
Source	Des Moines	Evansville	Fort Lauderdale
Daily Fixed Cost	$1,000	$3,000	$5,000
Destination	Albuquerque	Boston	Cleveland
Selling Price	$18	$30	$25

Supply Chain Planning

Executive Furniture Corporation (Supply Chain Planning)							
Using Source?	Des Moines	Evansville	Fort Lauderdale				Total Profit
Decision	1	1	0	(binary)			$ 5,600.00
Fixed Cost	$ 1,000.00	$ 3,000.00	$ 5,000.00				=
Cover Market?	Albuquerque	Boston	Cleveland				Total Revenue
Decision	0	1	1	(binary)			$ 11,000.00
Selling Price	$ 18.00	$ 30.00	$ 25.00				-
							Total Cost
Model Parameters		Des Moines	Evansville	Fort Lauderdale	Demand		$ 5,400.00
	Albuquerque	$ 5.00	$ 8.00	$ 9.00	300		
	Boston	$ 4.00	$ 4.00	$ 7.00	200		
	Cleveland	$ 3.00	$ 3.00	$ 5.00	200		
	Supply	100	300	300			
Source		Des Moines	Evansville	Fort Lauderdale	Demand Sum		Planned Demand
Destination	Albuquerque	0	0	0	0	=	0
	Boston	100	100	0	200	=	200
	Cleveland	0	200	0	200	=	200
	Supply Sum	100	300	0			
		≤	≤	≤			
	Planned Supply	100	300	0			

Shortest Path Problem





Shortest Path Problem

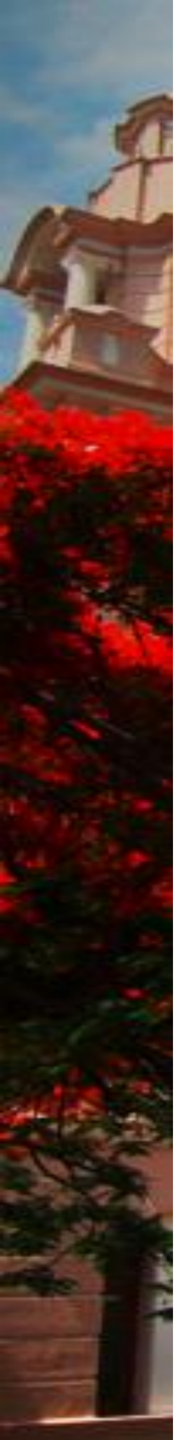
- Suppose we have real-time data of time to travel directly between any two addresses on the map.
- For simplicity, assume there are ten addresses.

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
A1	10000	2	5	8	9	11	13	20	10000	10000
A2	2	10000	10000	5	8	10000	25	35	50	10000
A3	4	10000	10000	6	11	18	30	10000	44	10000
A4	7	5	7	10000	8	10000	11	10000	10000	10000
A5	8	8	10	7	10000	10	18	5	8	44
A6	11	10000	15	10000	12	10000	3	9	10000	25
A7	14	22	33	12	19	1	10000	5	15	10000
A8	17	38	10000	10000	4	7	9	10000	6	14
A9	10000	44	55	10000	7	10000	18	9	10000	8
A10	10000	10000	10000	10000	50	28	10000	18	10	10000

Shortest Path Problem

- Suppose we need to go from A1 to A10. What is the fastest route?

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	Exit	
A1	0	0	0	0	1	0	0	0	0	0	1	
A2	0	0	0	0	0	0	0	0	0	0	0	
A3	0	0	0	0	0	0	0	0	0	0	0	
A4	0	0	0	0	0	0	0	0	0	0	0	
A5	0	0	0	0	0	0	0	0	1	0	1	
A6	0	0	0	0	0	0	0	0	0	0	0	
A7	0	0	0	0	0	0	0	0	0	0	0	
A8	0	0	0	0	0	0	0	0	0	0	0	
A9	0	0	0	0	0	0	0	0	0	1	1	
A10	0	0	0	0	0	0	0	0	0	0	0	
Enter	0	0	0	0	1	0	0	0	1	1		
Exit	1	0	0	0	1	0	0	0	1	0		
Starting	1	0	0	0	0	0	0	0	0	0		
Ending	0	0	0	0	0	0	0	0	0	1		
Balance	0	0	0	0	0	0	0	0	0	0		
	=	=	=	=	=	=	=	=	=	=		Total Time
	0	0	0	0	0	0	0	0	0	0		25

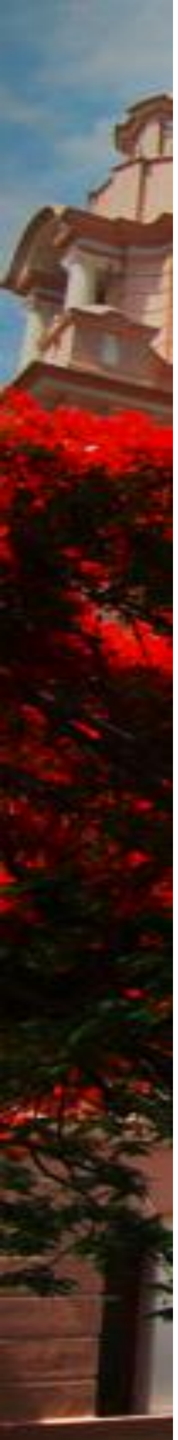


Large Scale Problems

- For the truck loading problem, suppose there are eighteen items and fifteen trucks. Excel cannot solve it because there are too many variables. We need to use Python and optimization packages.

Item	1	2	3	4	5	6	7	8	9
Weight	7500	7500	3000	3500	4000	3500	3000	3500	4000
Item	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
Weight	7500	3000	3500	7500	7500	3000	4000	3500	3000

Truck	1	2	3	4	5	6	7	8	9
Capacity	10000	5000	12000	8000	4500	4000	10000	5000	12000
Truck	**10**	**11**	**12**	**13**	**14**	**15**			
Capacity	4500	4000	10000	8000	4500	4000			



Large Scale Problems: Python + DoCplex

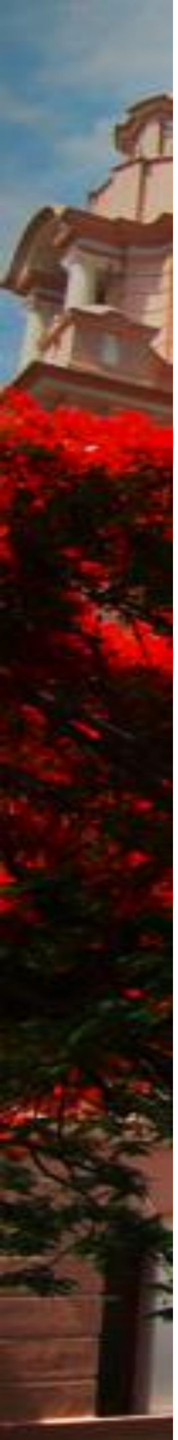
```
from docplex.mp.model import Model
nItem = 18; nTruck = 15
weight = [7500,7500,3000,3500,4000,3500,3000,3500,4000,7500,3000,3500,7500,7500,3000,4000,3500,3000]
capacity = [10000,5000,12000,8000,4500,4000,10000,5000,12000,4500,4000,10000,8000,4500,4000]
items = range(1,nItem+1); trucks = range(1,nTruck+1)

TLP = Model(name="TruckLoadingProblem")

#x[i,j] indicates whether to use truck i to load item j
x = {(i,j): TLP.binary_var(name="x_{0}_{1}".format(i,j)) for i in trucks for j in items}

#y[i] indicates whether to use truck i
y = {i: TLP.binary_var(name="y_{0}".format(i)) for i in trucks}

#The objective is to minimize unused capacity
TLP.minimize(TLP.sum(y[i]*capacity[i-1]-TLP.sum(x[i,j]*weight[j-1] for j in items) for i in trucks))

#Constraints: Load all the items
for j in items:
    TLP.add_constraint(TLP.sum(x[i,j] for i in trucks)==1)

#Constraints: Load <= Capacity limit
for i in trucks:
    TLP.add_constraint(TLP.sum(x[i,j]*weight[j-1] for j in items)<=y[i]*capacity[i-1])

TLPS = TLP.solve()
TLPS.display()
```

Large Scale Problems: Python + DoCplex

```
solution for: TruckLoadingProblem
objective: 9000
x_15_5 = 1
x_11_15 = 1
x_1_8 = 1
x_1_11 = 1
x_12_7 = 1
x_5_6 = 1
x_12_9 = 1
y_3 = 1
x_9_2 = 1
y_5 = 1
y_6 = 1
y_7 = 1
y_1 = 1
y_9 = 1
x_12_18 = 1
y_11 = 1
y_12 = 1
y_13 = 1
x_9_12 = 1
y_15 = 1
x_6_4 = 1
x_1_17 = 1
y_14 = 1
x_13_14 = 1
x_3_3 = 1
y_4 = 1
x_7_1 = 1
x_3_13 = 1
x_14_16 = 1
x_4_10 = 1
```

- CPLEX can solve the problem in less than a second.
- Note: variables with a zero value are omitted.
- Note: linear programming relaxation will not generate a meaningful lower bound here.

We will never have wasted capacity if we relax the integer constraints.



Large Scale Problems: Python + PuLP

```
nItem = 18; nTruck = 15
weight = [7500,7500,3000,3500,4000,3500,3000,3500,4000,7500,3000,3500,7500,7500,3000,4000,3500,3000]
capacity = [10000,5000,12000,8000,4500,4000,10000,5000,12000,4500,4000,10000,8000,4500,4000]
items = range(1,nItem+1); trucks = range(1,nTruck+1)

from pulp import *

TLP = LpProblem("TruckLoadingProblem",LpMinimize)

#x[(i,j)] indicates whether to use truck i to load item j
x = LpVariable.dicts("Assignments", [(i,j) for i in trucks for j in items], 0, 1, LpBinary)

#y[i] indicates whether to use truck i
y = LpVariable.dicts("TruckUsage", trucks, 0, 1, LpBinary)

#The objective is to minimize the total unused capacity
TLP += lpSum(y[i]*capacity[i-1]-lpSum(x[(i,j)]*weight[j-1] for j in items) for i in trucks)

#Constraints: Load all the items
for j in items:
    TLP += lpSum(x[(i,j)] for i in trucks) == 1

#Constraints: Load <= Capacity limit
for i in trucks:
    TLP += lpSum(x[(i,j)]*weight[j-1] for j in items) <= y[i]*capacity[i-1]

TLP.solve()

TOL = 0.0001
for i in trucks:
    if y[i].varValue > TOL:
        print("Truck ", i, " is used. The items are:")
        for j in items:
            if x[(i,j)].varValue > TOL:
                print(j)
print("Total unused capacity =",value(TLP.objective))
```

Large Scale Problems: Python + PuLP

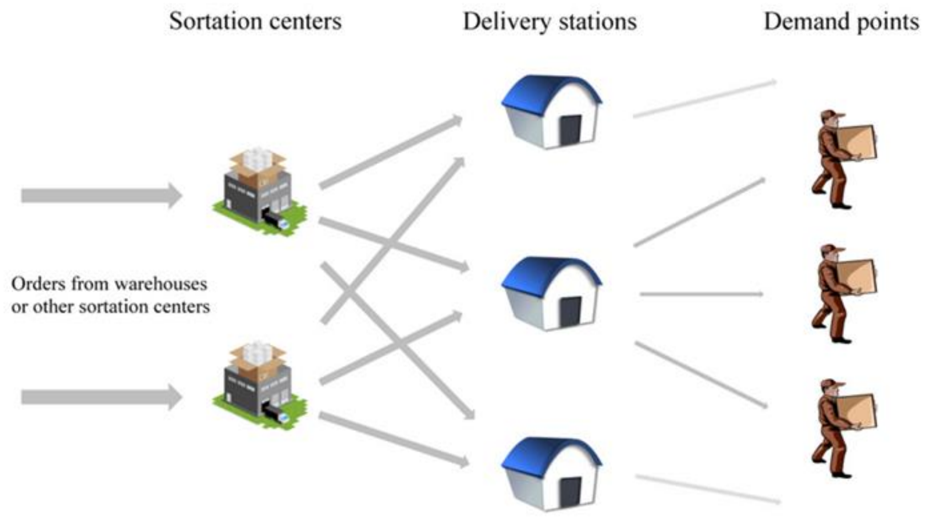
- The solution is different, but the objective value is the same. It means that the optimal solution is not unique.

```
Truck  1  is used. The items are:
2
Truck  3  is used. The items are:
1
6
Truck  4  is used. The items are:
14
Truck  5  is used. The items are:
5
Truck  6  is used. The items are:
11
Truck  7  is used. The items are:
3
4
15
Truck  9  is used. The items are:
8
10
Truck  11  is used. The items are:
7
Truck  12  is used. The items are:
12
17
18
Truck  13  is used. The items are:
13
Truck  14  is used. The items are:
9
Truck  15  is used. The items are:
16
Total unused capacity = 9000.0
```

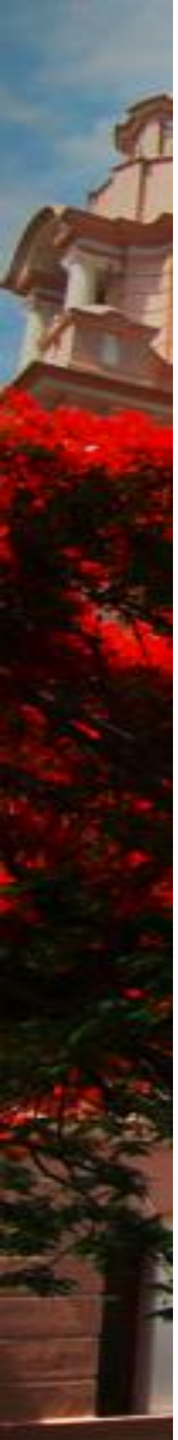
Case: JD.com Improves Delivery Networks

- Background
  - As JD.com expands its market presence and experiences increased demand, it needs to improve its delivery system to meet customer service targets while reducing operational costs.
- Objective
  - To develop a tool that can automatically generate annual plans for adjusting DS locations monthly and minimize total yearly operational costs.
- Challenges
  - …

Figure 1. (Color online) A Tripartite Graph Configuration of the Citywide Distribution Network of JD.com



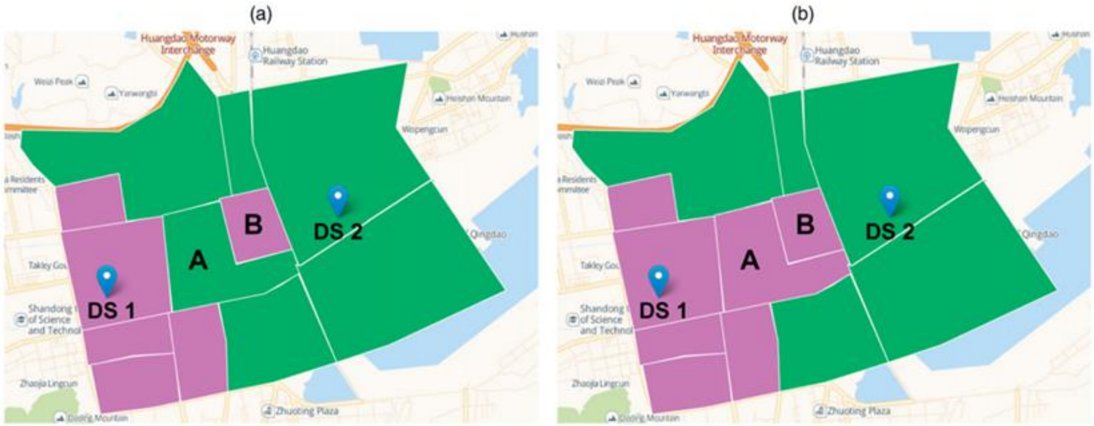
Note. The distribution network of JD.com consists of three layers of vertices, namely, sortation centers (SCs), delivery stations (DSs), and blocks (represented by demand points), as well as transportation arcs between SCs and DSs, and between DSs and blocks.



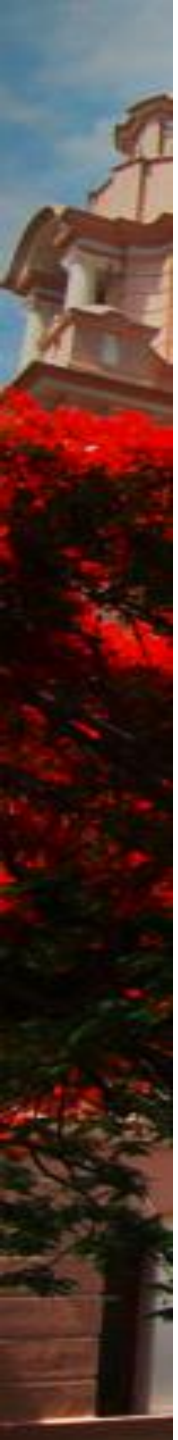
Case: JD.com Improves Delivery Networks

- Practical Restrictions
  - DS location changes: upper limit per month + monotonicity
  - Block assignment changes: upper limit per block
  - Connectivity requirement

Figure 3. (Color online) Two Different Plans of Block Assignments for Two Delivery Stations



Notes. The figures illustrate two different plans of block assignments for two delivery stations (DS): DS1 and DS2. In the left figure, the blocks assigned to DS1 are not connected, because block B is not adjacent to any other block covered by DS1. In the right figure, the blocks assigned to each of DS1 and DS2 are connected, which satisfies JD.com's connectivity requirements.



Case: JD.com Improves Delivery Networks

- Neighborhood-Restricted Heuristic

Table 1. Computation Performance Results for the Neighborhood-Based Heuristic with the Nested Warm-Start Algorithm in the JD.com Network in Qingdao

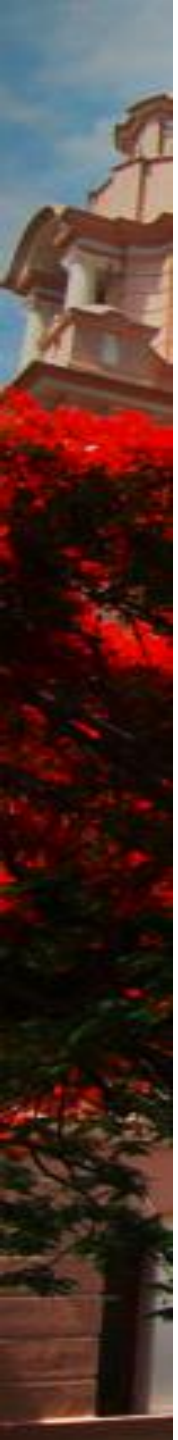
Relative size, τ[a]	Marginal time (seconds)[b]	Accumulative time (seconds)[c]	Heuristic gap (%)
1	34.61	34.61	19.911
2	46.93	81.54	8.985
3	131.25	212.79	4.210
4	8,591.14	8,803.93	2.250
5	28,803.17	37,607.10	1.251
6	1,396.80	39,003.90	0.338
7	1,856.73	40,860.63	0.259
8	6,068.74	46,929.37	0.067
9	1,087.30	48,016.67	0.063
10	1,859.24	49,875.91	0.062
20	10,437.88	60,313.79	0.014
50	9,792.06	70,105.85	0.002
100	18,335.57	88,441.42	0.000

[a]Relative size of restricted DS neighborhood.
[b]The computation time of solving the current problem by using the optimal solution of the former problem as the warm-start solution.
[c]The total computation time of solving the current problem.

- Model Results

Table 2. Plan Comparison for Qingdao in 2018

	Manual plan	Model plan (% change)	Benchmark plan (% change)
Total cost (CNY)	44,567,733	40,967,948 (–8.08%)	37,235,276 (–16.45%)
Fixed cost (CNY)	14,270,108	10,927,427 (–23.42%)	7,333,800 (–48.61%)
Delivery cost (CNY)	30,297,625	30,040,521 (–0.85%)	29,901,476 (–1.31%)
DS number in the last month	54	29 (–46.30%)	28 (–48.15%)
DS capacity use rate	37.14%	48.70% (11.56 pp)	74.18% (37.04 pp)
Average delivery distance (km)	5.53	5.33 (–3.61%)	5.22 (–5.58%)



Case: JD.com Improves Delivery Networks

- How to formulate the fixed cost of operating a DS at block j in month t, if we ignore the restrictions on the initial conditions, the number of changes, number of DSs, and the monotonicity and connectivity requirements?

Decision Variables

$$X_{tj} = \begin{cases} 1 & \text{if a DS exists at block } j \text{ in period } t, \\ 0 & \text{otherwise,} \end{cases} \quad t \in T \cup \{0\}, j \in J$$

$$Y_{tjk} = \begin{cases} 1 & \text{if block } k \text{ is assigned to DS } j \text{ in period } t, \\ 0 & \text{otherwise,} \end{cases} \quad t \in T \cup \{0\}, j, k \in J$$

$S_{tj} \geq 0$: DS size of block j in month t, $t \in T \cup \{0\}, j \in J$

$Z_{tj} \geq 0$: DS cost of block j in month t, $t \in T \cup \{0\}, j \in J$

Parameters

q_{tj} : number of packages to be delivered to block j at month $t, t \in T, j \in J$

α^{cap} : constant term of station capacity-DS size function

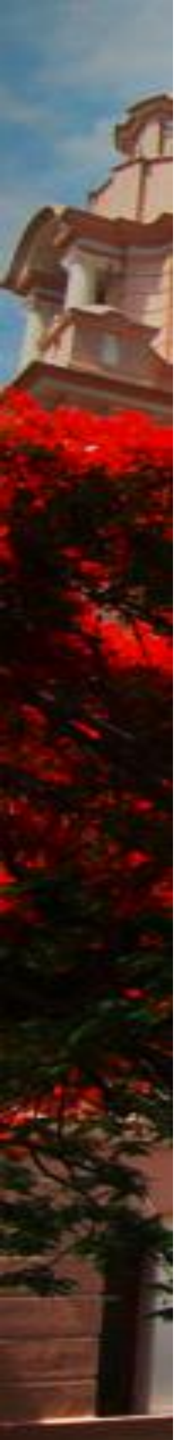
β^{cap} : coefficient of DS size in station capacity-DS size function

α^{cst} : constant term of the station cost

β_j^{cst} : unit DS size cost of block j (in RMB / month $/m^2$), $j \in J$

s_{0j} : initial DS size of block j; if there is no initial DS at block $j, s_{0j} = 0, j \in J$

s^{lb}, s^{ub} : lower bound and upper bound of size of new DSs



LP Relaxation

- When our problem has too many 0-1 variables, it will take too long to solve the problem. Sometimes, we may want to relax some or all of the 0-1 variables and allow them to take real values in [0,1] interval. That is, for a constraint of the form

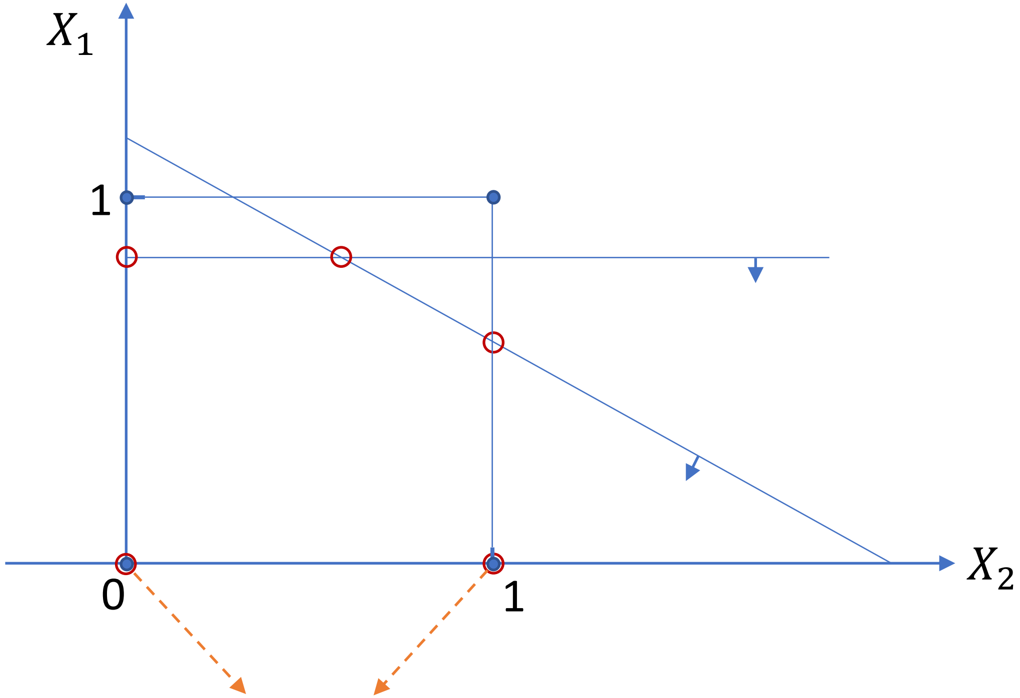
$$x_i \in \{0,1\},$$

- We replace it with a pair of linear constraints

$$0 \leq x_i \leq 1.$$

- What is the impact of LP relaxation? What will happen to our objective value?
- How can we make use of LP relaxation?

LP Relaxation

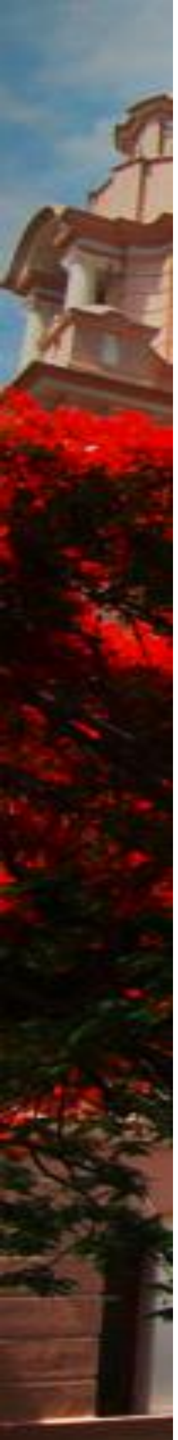


LP relaxation does not affect
these optimal solutions

Shortest Path Problem

- Now suppose for the shortest path problem there are tolls on some roads and we only have \$50. What is the shortest path under the budget constraint?

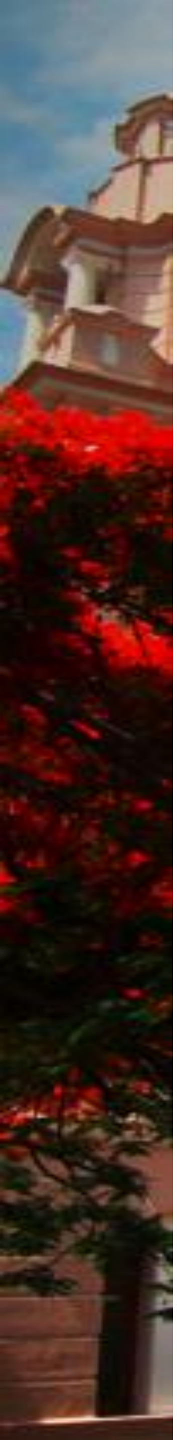
Cost	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
A1	10000	0	0	0	0	5	5	5	10000	10000
A2	0	10000	10000	0	0	10000	0	0	20	10000
A3	0	10000	10000	0	0	0	0	10000	20	10000
A4	0	0	0	10000	0	10000	20	10000	10000	10000
A5	0	0	0	0	10000	0	0	70	70	70
A6	5	10000	0	10000	0	10000	0	0	10000	20
A7	5	0	0	20	0	0	10000	0	20	10000
A8	5	0	10000	10000	70	0	0	10000	20	20
A9	10000	20	20	10000	70	10000	20	20	10000	0
A10	10000	10000	10000	10000	70	20	10000	20	0	10000



Shortest Path Problem

- The solution under LP relaxation:

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	Exit	
A1	0	0	0	0	0.555556	0	0.444444	0	0	0	1	
A2	0	0	0	0	0	0	0	0	0	0	0	
A3	0	0	0	0	0	0	0	0	0	0	0	
A4	0	0	0	0	0	0	0	0	0	0	0	
A5	0	0	0	0	0	0	0	0	0.555556	0	0.555556	
A6	0	0	0	0	0	0	0	0	0	0	0	
A7	0	0	0	0	0	0	0	0.444444	0	0	0.444444	
A8	0	0	0	0	0	0	0	0	0.444444	0	0.444444	
A9	0	0	0	0	0	0	0	0	0	1	1	
A10	0	0	0	0	0	0	0	0	0	0	0	
Enter	0	0	0	0	0.555556	0	0.444444	0.444444	1	1		Total Cost
Exit	1	0	0	0	0.555556	0	0.444444	0.444444	1	0		50
Starting	1	0	0	0	0	0	0	0	0	0		<=
Ending	0	0	0	0	0	0	0	0	0	1		50
Balance	0	0	0	0	0	0	0	0	0	0		
	=	=	=	=	=	=	=	=	=	=		Total Time
	0	0	0	0	0	0	0	0	0	0		28.11111



Shortest Path Problem

- The solution after rounding (to the nearest integer such that the budget constraint is satisfied):

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	Exit	
A1	0	0	0	0	0	0	1	0	0	0	1	
A2	0	0	0	0	0	0	0	0	0	0	0	
A3	0	0	0	0	0	0	0	0	0	0	0	
A4	0	0	0	0	0	0	0	0	0	0	0	
A5	0	0	0	0	0	0	0	0	0	0	0	
A6	0	0	0	0	0	0	0	0	0	0	0	
A7	0	0	0	0	0	0	0	1	0	0	1	
A8	0	0	0	0	0	0	0	0	1	0	1	
A9	0	0	0	0	0	0	0	0	0	1	1	
A10	0	0	0	0	0	0	0	0	0	0	0	
Enter	0	0	0	0	0	0	1	1	1	1		Total Cost
Exit	1	0	0	0	0	0	1	1	1	0		25
Starting	1	0	0	0	0	0	0	0	0	0		<=
Ending	0	0	0	0	0	0	0	0	0	1		50
Balance	0	0	0	0	0	0	0	0	0	0		
	=	=	=	=	=	=	=	=	=	=		Total Time
	0	0	0	0	0	0	0	0	0	0		32

It is indeed the optimal solution in this case!

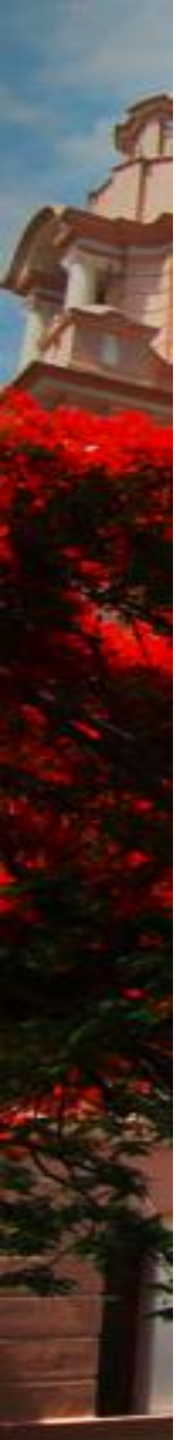
LP Relaxation and Bounds

- A mixed integer program can be written in the following generic form

$$y^* = \max_{X,Z} \; c'X + f'Z$$

$$\begin{aligned} s.t. \quad AX + BZ &\leq b \\ X &\geq 0 \\ Z_i &\in \{0,1\} \end{aligned}$$

- If the integer constraint for Z is relaxed, let $(\tilde{X}, \tilde{Z})$ be the optimal solution under LP relaxation and $\tilde{y}$ be the optimal value.
- Suppose we can round each component $\tilde{Z}_i$ to the nearest integer such that all the constraints are satisfied. Let $\hat{Z}$ be the vector of rounded values and $\hat{y}$ be the corresponding objective value.



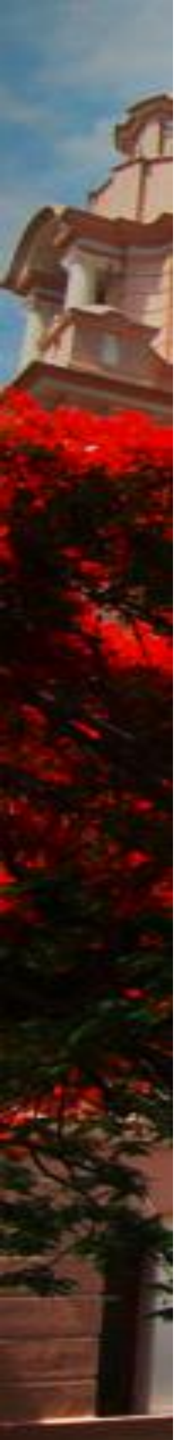
LP Relaxation and Bounds

- We have

$$\hat{y} \leq y^* \leq \tilde{y}$$

- Sketch of proof:
- By definition, $(\tilde{X}, \hat{Z})$ is a feasible solution and thus $\hat{y} \leq y^*$.
- Suppose (X^*, Z^*) is the optimal solution for the original problem.
- Clearly, (X^*, Z^*) is a feasible solution for the LP-relaxed problem.
- By definition, $(\tilde{X}, \tilde{Z})$ is the optimal solution for the relaxed problem and objective function is unchanged in the relaxation. Hence, $y^* \leq \tilde{y}$ from the perspective of the relaxed problem.

- Therefore, LP relaxation can generate an upper bound and rounding of the relaxed solution can generate a lower bound for the original problem.



In-Class Exercise

- Suppose in the supply chain planning problem, each plant, if used, must use at least 80% of its capacity. How to model this constraint?

- Suppose it is required that a single plant cannot supply multiple destinations. How to model this constraint and modify the model?

- Suppose in the shortest path problem we are required to pass by node 4. How to incorporate this constraint?