Optimization Theory and Methods

2025 Autumn

同济经管 TONGJI SEM

魏可佶 kejiwei@tongji.edu.cn
https://kejiwei.github.io/

- Modeling integer programming problems (IPs)
- Warehouse location example
- Forcing constraints, Big M-type constraints
- Branch-and-bound algorithm
- Optimal investment example

- A company is considering opening warehouses in up to 4 cities:
  - New York, Los Angeles, Chicago, and Atlanta.
  - Each warehouse can ship 100 units per week. The weekly fixed cost of keeping each warehouse open is $400 for New York, $500 for LA, $300 for Chicago, and $150 for Atlanta.
  - Region 1 requires 80 units per week, region 2 requires 70 units per week, and region 3 requires 40 units per week. The shipping costs are shown in the table below.

From\To	Region 1	Region 2	Region 3
New York	20	40	50
Los Angeles	48	15	26
Chicago	26	35	18
Atlanta	24	50	35

- Formulate the problem to meet weekly demand at minimum cost.

- **What are the decision variables?**
  - Whether or not to open a warehouse in a city, there are 4 cities.
  - Flow from a warehouse to a region: There are 4 possible warehouse locations and 3 regions. So 12 combinations.
- **What is the objective function?**
  - Minimize total cost = fixed cost + shipping cost.
- **What are the constraints?**
  - (1) Demand constraint, one for each region.
  - (2) Capacity constraint, one for each open warehouse.
  - (3) Additionally, we must ensure that no flow comes out of a non-existent warehouse!
- **We can combine 2 and 3 into what is called a forcing constraint.**

同济经管 TONGJI SEM

- Data:
  - C: set of cities, R: set of regions
  - c_i: weekly fixed cost of warehouse in city i
  - b_j: demand from region j
  - t_{ij}: shipping cost from warehouse in city i to region j
- Decision variables:
  - y_i: whether or not to open a warehouse in city i
  - x_{ij}: flow from warehouse in city i to region j

What Are Integer Optimization Problems?

$$\boldsymbol{MIN}\ \left(\sum_{i\in C} c_i * y_i + \sum_{i\in C}\sum_{j\in R} t_{ij} * x_{ij}\right)$$

s.t.

$$\sum_{j\in R} x_{ij} \le 100 * y_i, \forall i \in C$$
$$\sum_{i\in C} x_{ij} = b_j, \forall j \in R$$
$$x_{ij} \in \mathbb{Z}^+, \forall i \in C, j \in R$$
$$y_i \in \{0,1\}, \forall i \in C$$

- **Forcing constraint ensures that:**
  - Flow out of each open warehouse does not exceed its capacity ($y_i = 1$)
  - Flow out of warehouses, which are not open, is zero ($y_i = 0$)
- **What if each warehouse's capacity is infinite (sufficient)?**
  - Assume the capacity to be a sufficiently large number (M)
  - Any number ≥ 190 works in this case. (Why?)

$$\sum_{j \in R} x_{ij} \leq M * y_i, \forall i \in C$$

- **Can model OR type constraints:**

$$ax + by \leq c \text{ or } dx + ey \leq f, \quad ax + by \leq c + Mz$$

$$dx + ey \leq f + M(1 - z), z \in \{0, 1\}$$

↳ Warehouse Location : Additional Constraints

- If the New York warehouse is opened, the LA warehouse must be opened.

$$y_{NYC} \leq y_{LA}$$

- At most two warehouses can be opened.

$$\sum_{i \in C} y_i \leq 2$$

- EXOR constraint: Either Atlanta or LA warehouse must be opened but not both.

$$y_{LA} + y_{ATL} = 1$$

- If we open a warehouse in Chicago, then we may open warehouses on at most one coast. (East coast has NYC and ATL; west coast has LA.)

LP Relaxation of IPs

- LP relaxation of an IP is a formulation obtained by relaxing the integrality constraints in an IP.

$$\boldsymbol{x} \in \mathbb{Z}^{+} \rightarrow \boldsymbol{x} \geq \mathbf{0}$$
$$\boldsymbol{x} \in \{\mathbf{0}, \mathbf{1}\} \rightarrow \mathbf{0} \leq \boldsymbol{x} \leq \mathbf{1}$$

- If an optimal solution to the relaxed problem is feasible to the original IP/MIP/BIP (i.e., if the optimal solution to the relaxed problem is integral), then it is also an optimal solution to the IP.
- LP relaxation provides a lower (upper) bound on the optimal objective function value of an IP minimization (maximization) problem.
- Good formulations allow the LP relaxation to provide a “tight” bound on the IP.

- A smart enumeration algorithm.
- Branch-and-bound involves 3 major operations:
  - Branching enumerates all possible solutions in a tree structure.
  - Bounding provides the lower (upper) bounds on optimal objective function values in each branch for minimization (maximization) problems; bounds are usually based on LP relaxation, but other types of bounds are also possible.
  - Pruning ensures that only a small subset of the possible solutions are actually evaluated; many branches are trimmed prematurely.

- 3 types of possible reasons for pruning: *the 3 I's*

(1) Infeasible: If the LP relaxation of a branch leads to infeasible problem

- If LP relaxation is infeasible, the IP will also be infeasible. (Why?)

(2) Inferior to current best: If optimal LP solution is worse than current best

- If the optimal objective function value of the LP relaxation of a branch is inferior to the best available integer solution, then the IP solution of that branch will also be inferior to the best available integer solution.

(3) Integral: If the optimal solution of the LP relaxation is integral

- If the optimal solution to the LP relaxation of a branch turns out to be integral, then that is also the optimal IP solution of that branch. So we don't need to branch any further.

- An investor wants to invest the available cash ($100K) in a combination of four available alternatives. She needs to maximize the total expected returns while ensuring that the variance does not exceed a threshold (50 $(K\$)^2$) to contain the portfolio risk. Asset fractions cannot be selected.
- Assume that the returns on the four assets are independent of each other. So the variance of sum is the sum of variances.
- What assets should she invest in to maximize total expected return?

Item	Investment Amount (K $)	Expected Return ($)	Variance of Return $(K\$)^2$
Asset 1	20	5000	35
Asset 2	50	3000	22
Asset 3	50	2100	20
Asset 4	30	1500	17

- **Formulation**

$$MAX\ (5000x_1 + 3000x_2 + 2100x_3 + 1500x_4)$$

s.t.

$$20x_1 + 50x_2 + 50x_3 + 30x_4 \leq 100$$

$$35x_1 + 22x_2 + 20x_3 + 17x_4 \leq 50$$

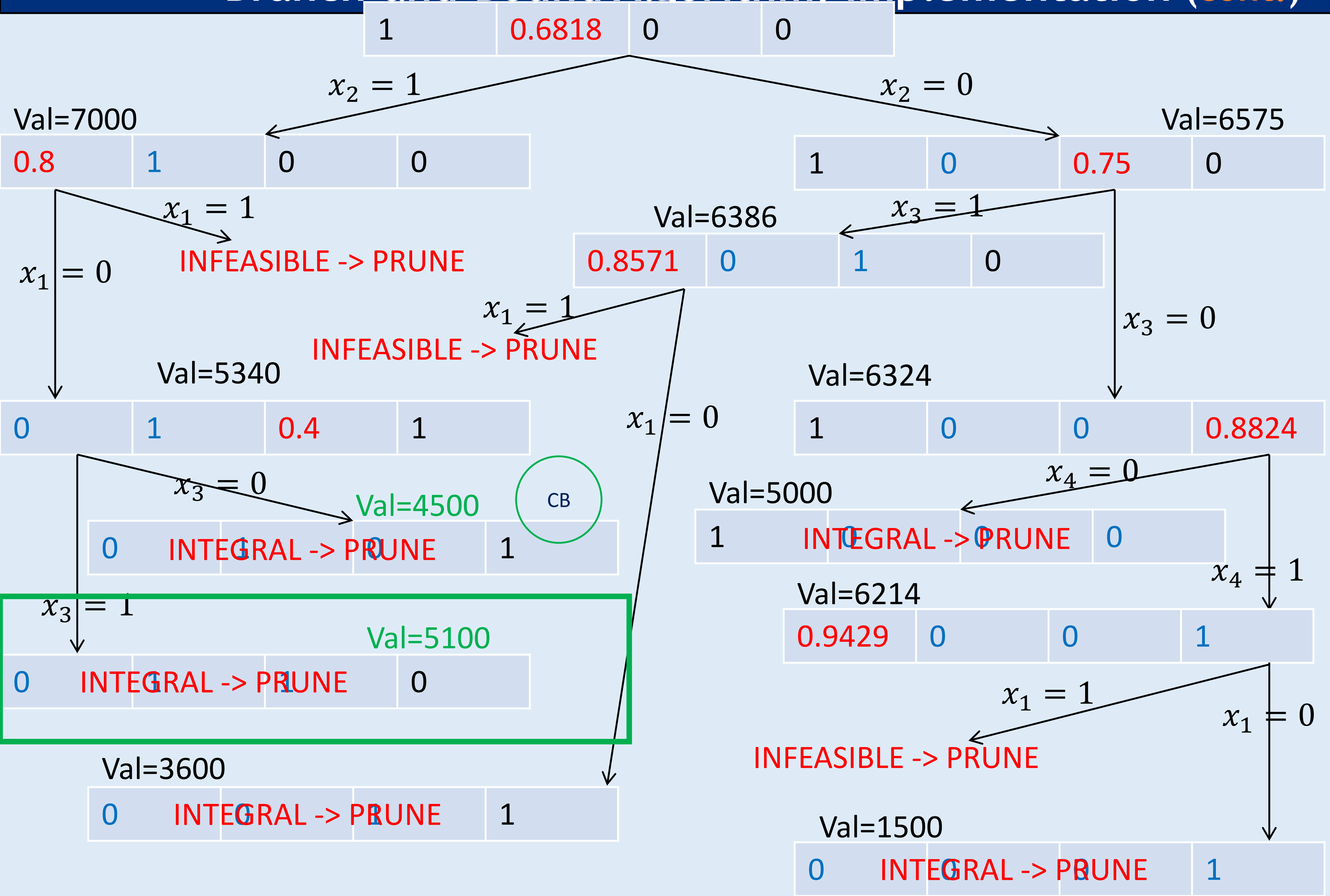
$$x_1, x_2, x_3, x_4 \in \{0,1\}$$

x_i: Binary variable that equals 1 if asset i is selected and 0 otherwise

Beginning with root node (minimization):

- Bound
  - Solve the current LP with this and all restrictions along the path from the root node enforced.
- Prune
  - If optimal LP value is greater than or equal to the current best solution => Prune.
  - If LP is infeasible => Prune.
  - If LP is integral => Prune.
- Branch
  - Set some variable to an integer value.
- Repeat until all nodes pruned.

↳ Branch-and-Bound Algorithm: Implementation (cont.)



Objective :

Key Concepts：