

Integer Programming

Taek-Ho Lee

Department of Industrial Engineering, SeoulTech

Mail: taekho.lee@seoultech.ac.kr

Introduction

- Limitations of the divisibility assumption in Linear Programming
 - In reality, many decision variables must be integers → Integer Programming
- Types of integer programming
 - Pure integer programming: All decision variables are integers.
 - ✓ (Pure) Binary integer programming (BIP): All decision variables are binary variables.
 - Mixed integer programming (MIP): Some of the decision variables are integers.

Prototype Example

	Decision variables	Current value (\$ million)	Required cost (\$ million)
Plant in LA	x_1	9	6
Plant in SF	x_2	5	3
Warehouse in LA	x_3	6	5
Warehouse in SF	x_4	4	2

- Background

- There are plans to build a new plant in either Los Angeles or San Francisco, or in both.
- In addition, the construction of at most one new warehouse is being considered.
 - ✓ A warehouse can only be built in a location where a plant is also built.
- The maximum amount of money to be spent, depending on the decision is 10(million \$).

Prototype Example

	Decision variables	Current value (\$ million)	Required cost (\$ million)
Plant in LA	x_1	9	6
Plant in SF	x_2	5	3
Warehouse in LA	x_3	6	5
Warehouse in SF	x_4	4	2

- BIP Model

- All decision variables are binary
 - ✓ $x_{ij} = 1$ if the j -th decision is yes, and $x_{ij} = 0$ otherwise.
- Present value of profit from the decisions: $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$
 - ✓ Maximum available budget constraint: $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$
 - ✓ Warehouse count constraint: $x_3 + x_4 \leq 1$
 - ✓ Warehouse requires a plant constraint:
 - $x_3 \leq x_1 \rightarrow x_3 - x_1 \leq 0$
 - $x_4 \leq x_2 \rightarrow x_4 - x_2 \leq 0$

Prototype Example

	Decision variables	Current value (\$ million)	Required cost (\$ million)
Plant in LA	x_1	9	6
Plant in SF	x_2	5	3
Warehouse in LA	x_3	6	5
Warehouse in SF	x_4	4	2

- BIP Model

- Maximize $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$
 - ✓ $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$
 - ✓ $x_3 + x_4 \leq 1$
 - ✓ $-x_1 + x_3 \leq 0$
 - ✓ $-x_2 + x_4 \leq 0$
 - ✓ $0 \leq x_j \leq 1$ (x_j is an integer) $\rightarrow x_j$ is a binary variable.

Perspectives on Solving IP Problems

- Essential Difficulty of Integer Programming
 - While LP can be solved efficiently thanks to the property that only Corner Point Feasible (CPF) solutions need to be considered, even though there may be infinitely many solutions, integer programming (IP) poses a much harder challenge.
 - Although IP has only finitely many solutions, the number of possible cases grows exponentially as the problem size increases. As a result, even relatively small problems can become very difficult to solve.
 - Therefore, it is important to recognize that no algorithm can solve IP quickly in general, purely as it is.

Perspectives on Solving IP Problems

- The Key Role of LP Relaxation
 - Accordingly, part of an IP problem can be linked to the corresponding LP problem (LP relaxation problem) so that the Simplex Method can be applied.
 - When solving the LP problem while ignoring the integer constraints of the IP, we obtain an optimal solution with fractional values. In this case, the objective function value serves as an upper bound for the optimal IP solution (for maximization problems).
 - However, one cannot simply round the LP relaxation solution to the nearest integer solution.
 - ✓ The nearest integer solution may not be feasible.
 - (ex) $\text{Max } Z = x_2$, (subject to) $-x_1 + x_2 \leq 0.5$, $x_1 + x_2 \leq 3.5$, and x_1, x_2 are integers.
 - ✓ The nearest integer solution may not be optimal.
 - (ex) $\text{Max } Z = x_1 + 5x_2$, (subject to) $x_1 + 10x_2 \leq 20$, $x_1 \leq 2$, and x_1, x_2 are integers.

The Branch-and-Bound Technique for BIP

- Branch-and-Bound Technique
 - The most representative exact method for solving IP problems.
 - Core idea
 - ✓ Divide-and-conquer: The original problem is divided into subproblems and solved.
 - ✓ Branching: Partitioning the entire set of feasible solutions.
 - ✓ Fathoming: Identifying bounds on the optimal solution for each subset and discarding subsets that cannot contain an optimal solution.
 - Three key steps
 - ✓ Branching: If the LP relaxation solution of the given problem is integer, then it is already optimal.
 - If not, branch the problem into two subproblems based on one of the non-integer variables.
 - ✓ Bounding: Solve each subproblem using LP to calculate the bound (objective value bound).
 - ✓ Fathoming
 - Discard branches when they are unnecessary.
 - (ex) If the bound of a subproblem is worse than the best known feasible solution, then that branch is fathomed (eliminated).

The Branch-and-Bound Technique for BIP

- Branch-and-Bound Technique: Algorithm Procedure
 - Initialization: Set $Z^* = -\infty$ and solve the LP relaxation of the original problem.
 - ✓ If the optimal solution of the LP relaxation is integer, terminate.
 - ✓ If the solution is fractional, perform branching.
 - Iteration
 - ✓ Branching: Select one variable that violates the integrality condition and create two subproblems.
 - ✓ Bounding: For each subproblem, solve the LP relaxation and compute the objective value.
 - ✓ Fathoming: discard the subproblem in the following cases
 - 1. The LP relaxation of the subproblem is infeasible
 - 2. The optimal value of the LP relaxation is worse than the best known integer feasible solution (Z^*) (for maximization problems: smaller than Z^*)
 - 3. The optimal solution of the LP relaxation is integer.
 - If it improves the current best known solution, update Z^* . Then repeat check (1) for all branches.
 - ✓ Optimality test: If no unexplored subproblems remain, terminate. The current Z^* is the optimal solution.

The Branch-and-Bound Technique for BIP

- Branch-and-Bound Technique – Example
 - Maximize $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$
 - ✓ $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$
 - ✓ $x_3 + x_4 \leq 1$
 - ✓ $-x_1 + x_3 \leq 0$
 - ✓ $-x_2 + x_4 \leq 0$
 - ✓ (All x_i are binary)
 - Initialization
 - ✓ LP relaxation problem ($0 \leq x_j \leq 1$ for all $j = 1, 2, 3, 4$)
 - Optimal solution $(x_1, x_2, x_3, x_4) = (5/6, 1, 0, 1)$ with objective $Z = 16.5$
 - Since some variables do not satisfy integrality, this is not optimal.
 - ✓ Bounding for the entire problem
 - Given the structure of the objective, Z must be an integer for any feasible solution, therefore $Z \leq 16$.

The Branch-and-Bound Technique for BIP

- Branch-and-Bound Technique – Example

- Branching

- ✓ For a binary variable x_j , partition the full feasible set as

- (1) $x_j = 1$, (2) $x_j = 0$

- ✓ Subproblem 1: $x_j = 0$

- Max $Z = 5x_2 + 6x_3 + 4x_4$

- $3x_2 + 5x_3 + 2x_4 \leq 10$

- $x_3 + x_4 \leq 1$

- $x_3 \leq 0$

- $-x_2 + x_4 \leq 0$

- ✓ Subproblem 2: $x_j = 1$

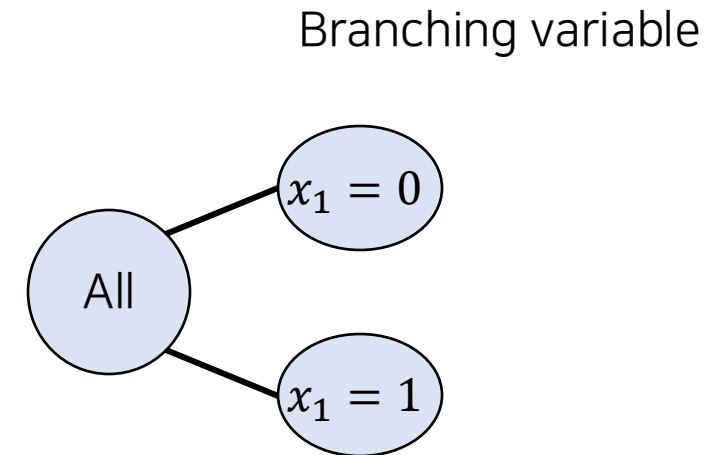
- Max $Z = 9 + 5x_2 + 6x_3 + 4x_4$

- $3x_1 + 5x_3 + 2x_4 \leq 4$

- $x_3 + x_4 \leq 1$

- $x_3 \leq 1$

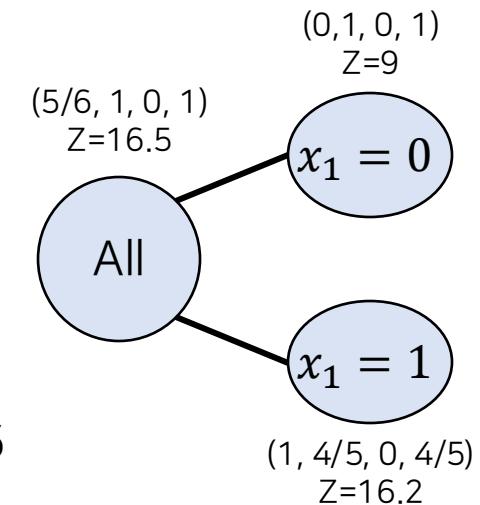
- $-x_2 + x_4 \leq 0$



Branching tree
or Solution tree
or Enumeration tree

The Branch-and-Bound Technique for BIP

- Branch-and-Bound Technique – Example
 - Bounding
 - ✓ Quickly explore how good the best feasible solution of each subproblem could be by solving its LP relaxation.
 - LP relaxation: replace each binary variable with $0 \leq x_j \leq 1$
 - ✓ Subproblem 1 ($x_1=0$)
 - Optimal solution of the LP relaxation: $(0, 1, 0, 1)$ with $Z = 9$
 - Best objective value found so far: $Z^*=9$
 - ✓ Subproblem 2 ($x_1=1$)
 - Optimal solution of LP relaxation: $(1, 4/5, 0, 4/5)$ with $Z = 16.2$
 - Further subproblems under this branch have upper bound $Z \leq 16$



The Branch-and-Bound Technique for BIP

▪ Branch-and-Bound Technique – Example

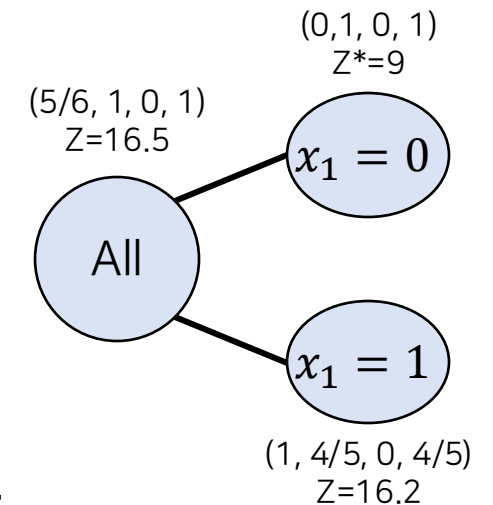
• Fathoming

✓ Subproblem 1 ($x_1=0$)

- The LP relaxation solution $(0, 1, 0, 1)$ is already an integer solution \rightarrow first candidate solution.
- Since the subproblem has produced an integer solution, no further branching is needed \rightarrow fathomed.
- Also, since the current best $Z^*=9$, any future subproblems whose bound is less than or equal to 9 can also be fathomed.

✓ Fathoming test

- 1. The bound of the subproblem \leq the best known integer Z^*
- 2. The LP relaxation of the subproblem is infeasible
- 3. The LP relaxation of the subproblem yields an integer solution.
 - In case (3), if the new integer solution is better than the current Z^* , update Z^* and then re-apply fathoming test (1) to all remaining subproblems.



The Branch-and-Bound Technique for BIP

▪ Branch-and-Bound Technique – Example

• Iteration 2.

✓ Subproblem 3: $x_1=1, x_2=0$

– Maximize $Z = 9 + 6x_3 + 4x_4$

• $5x_3 + 2x_4 \leq 4$

• $x_3 + x_4 \leq 1$

• $x_3 \leq 1$

• $x_4 \leq 0$

✓ Subproblem 4: $x_1=1, x_2=1$

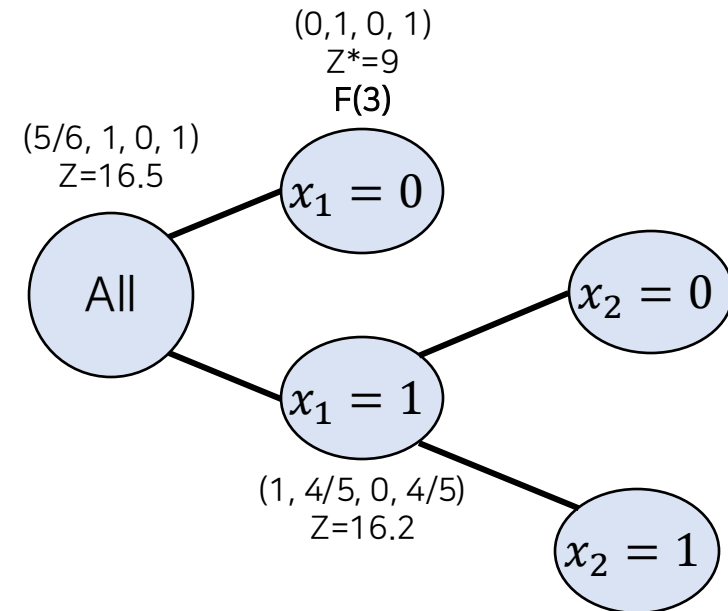
– Maximize $Z = 14 + 6x_3 + 4x_4$

• $5x_3 + 2x_4 \leq 1$

• $x_3 + x_4 \leq 1$

• $x_3 \leq 1$

• $x_4 \leq 1$



The Branch-and-Bound Technique for BIP

- Branch-and-Bound Technique – Example

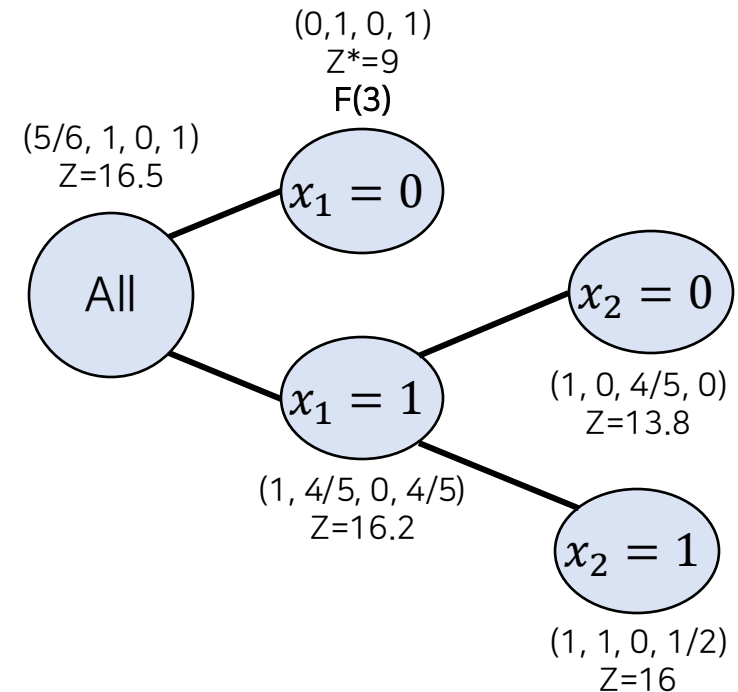
- Iteration 2.

- ✓ Subproblem 3: $x_1=1, x_2=0$

- LP relaxation optimum = $(1, 0, 4/5, 0)$ with $Z = 13.8$

- ✓ Subproblem 4: $x_1=1, x_2=1$

- LP relaxation optimum = $(1, 1, 0, 1/2)$ with $Z = 16$



The Branch-and-Bound Technique for BIP

▪ Branch-and-Bound Technique – Example

• Iteration 3.

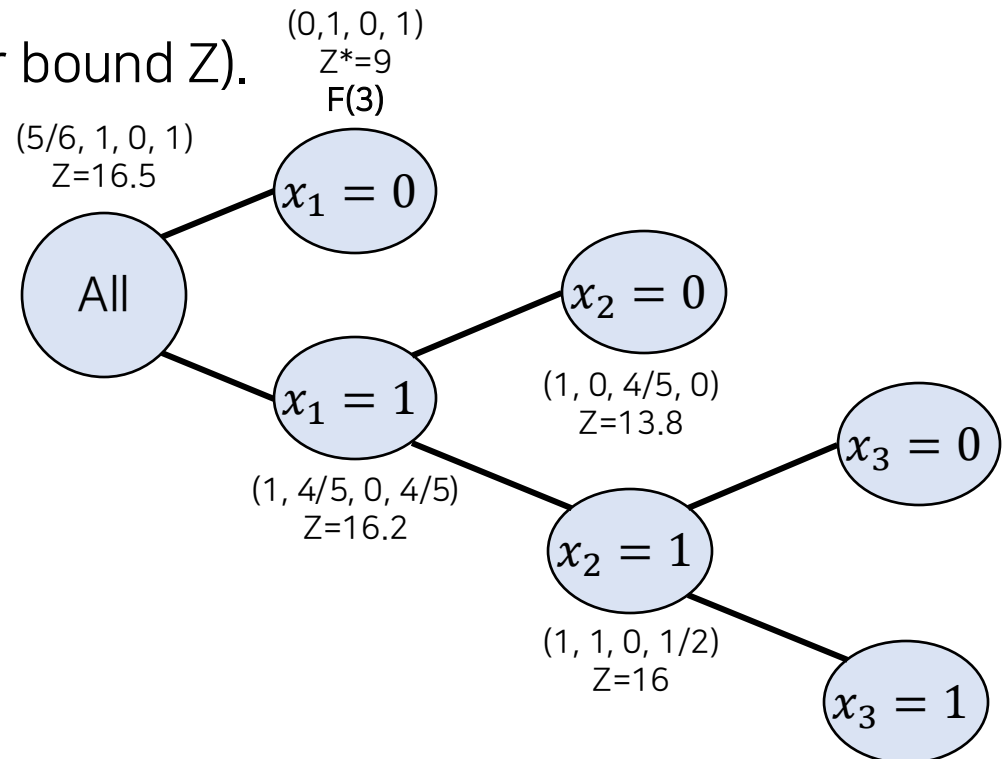
- ✓ When multiple remaining subproblems exist, choose the most recently generated subproblem (if tied, choose the one with the larger upper bound Z).

- ✓ Subproblem 5: $x_1=1, x_2=1, x_3=0$

- Maximize $Z = 14 + 4x_4$
 - $2x_4 \leq 1$
 - $x_4 \leq 1$

- ✓ Subproblem 6: $x_1=1, x_2=1, x_3=1$

- Maximize $Z = 20 + 4x_4$
 - $2x_4 \leq -4$
 - $x_4 \leq 0$
 - $x_4 \leq 1$

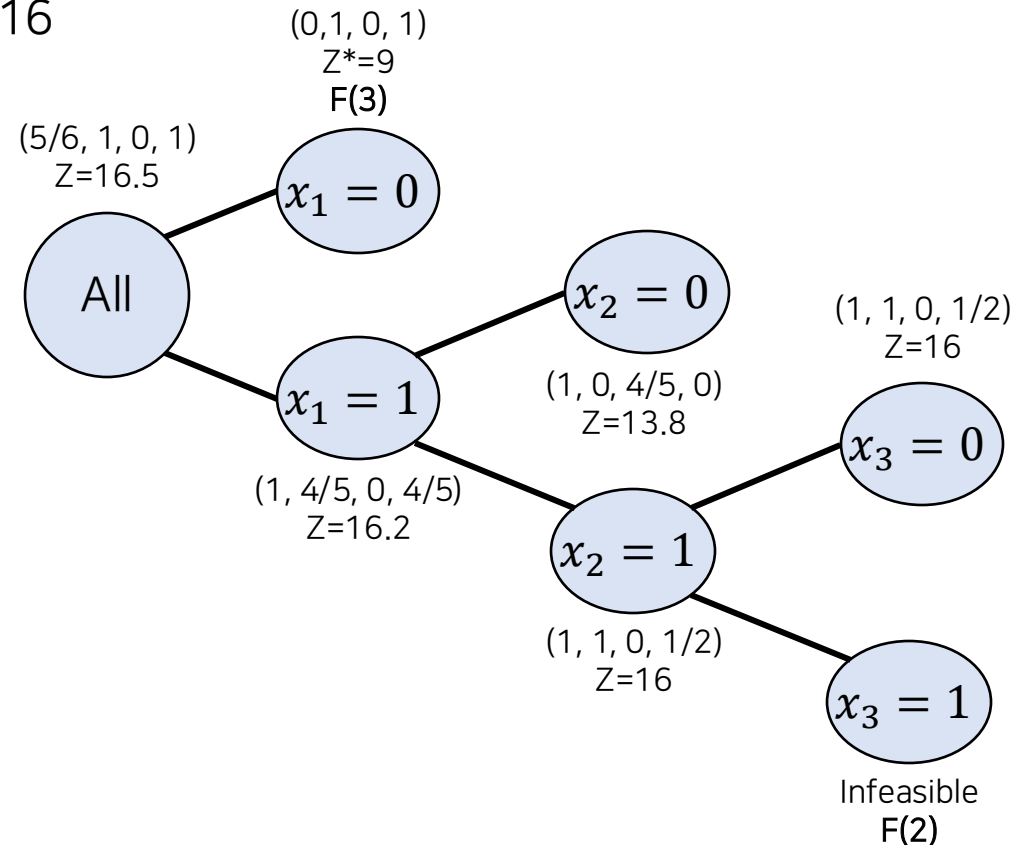


The Branch-and-Bound Technique for BIP

- Branch-and-Bound Technique – Example

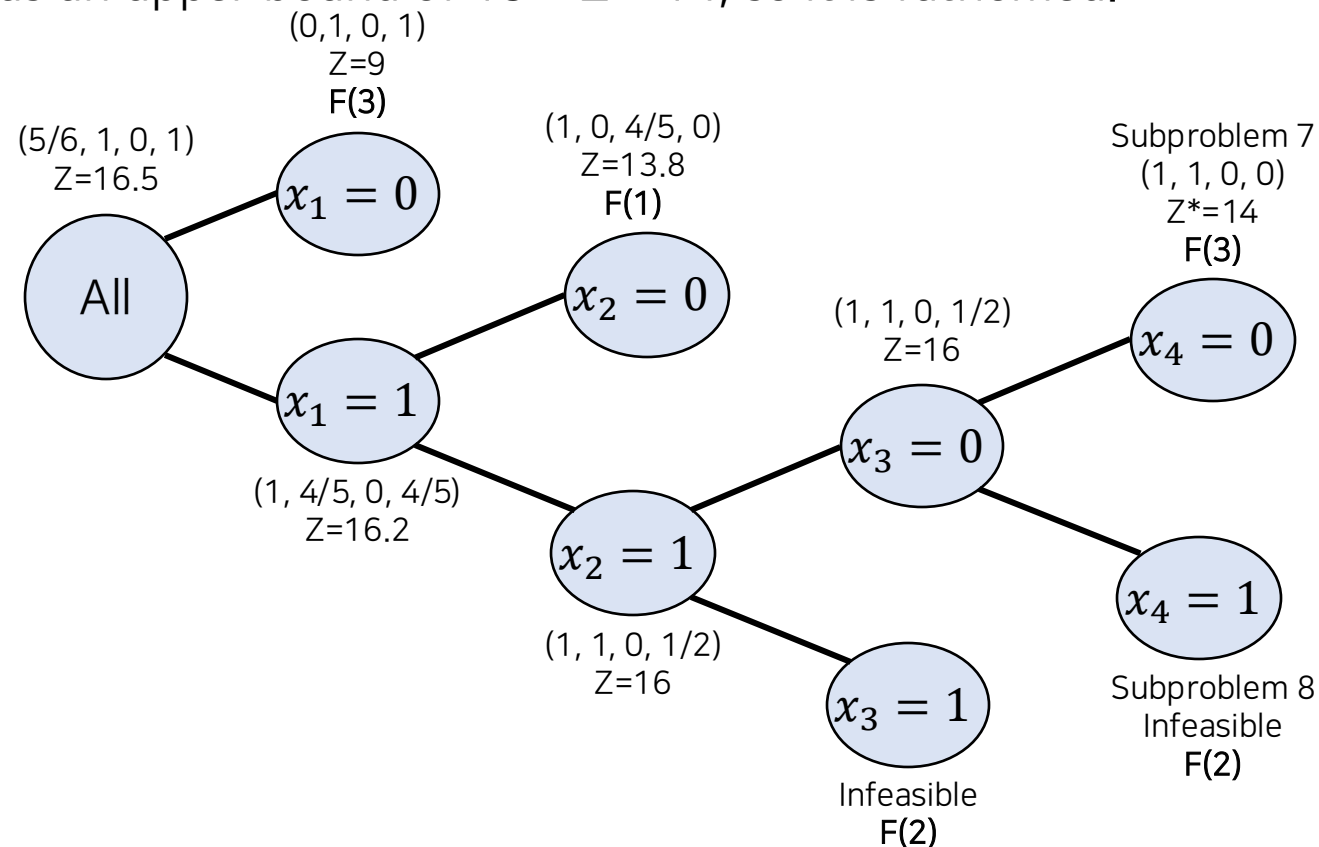
- Iteration 3.

- ✓ Subproblem 5: $x_1=1, x_2=1, x_3=0$
 - LP relaxation optimum = $(1, 1, 0, 1/2)$ with $Z = 16$
- ✓ Subproblem 6: $x_1=1, x_2=1, x_3=1$
 - LP relaxation is infeasible.



The Branch-and-Bound Technique for BIP

- Branch-and-Bound Technique – Example
 - Iteration 4.
 - ✓ The remaining branches are no longer subproblems but are themselves integer solutions.
 - ✓ Also, subproblem 3 has an upper bound of $13 \leq Z^* = 14$, so it is fathomed.



The Branch-and-Bound Technique for BIP

- Branch-and-Bound Technique – Other Options
 - The branch-and-bound technique is not a single algorithm but rather a general framework with high flexibility.
 - Branching refers to selecting one residual subproblem and dividing it into smaller subproblems.
 - ✓ Rules for selecting which residual subproblem to branch on:
 - Most recent subproblem first: Easier to reoptimize using the latest Simplex Method results.
 - Subproblem with the best bound first: Higher chance of finding a better Z^* , which allows pruning the rest.
 - Other approaches include Depth-First Search (DFS), Breadth-First Search (BFS), and Best-First Search.
 - ✓ Rules for dividing a selected subproblem into smaller subproblems:
 - Typically involves choosing a branching variable and splitting based on its value or range.

The Branch-and-Bound Technique for BIP

- Branch-and-Bound Technique – Other Options
 - The branch-and-bound technique is not a single algorithm but rather a general framework with high flexibility.
 - Bounding: Performed by solving a relaxation problem of the given subproblem.
 - ✓ The relaxation problem can be formulated not only by simply removing integer constraints and turning it into an LP, but also through various other methods.
 - The main criteria are (1) relaxation methods that can be solved quickly, (2) relaxation methods that usually provide good bounds.
 - ✓ Examples include Lagrangian relaxation and others.

The Branch-and-Bound Technique for MIP

- Branch-and-Bound Technique – Application to MIP
 - For convenience of representation in MIP, assume that among a total of n variables are integers.
 - Maximize $Z = \sum_{j=1}^n c_j x_j$
 - ✓ $\sum_{j=1}^n a_{ij} x_j \leq b_i$, for $i = 1, \dots, n$
 - ✓ $x_j \geq 0$, for $i = 1, \dots, n$
 - ✓ x_j is integer, for $i = 1, \dots, I$

The Branch-and-Bound Technique for MIP

- Branch-and-Bound Technique for MIP
 - Change from pure binary integer programming to mixed integer programming
 - ✓ Binary variable \rightarrow Integer variable
 - ✓ Pure IP \rightarrow Mixed IP
 - Changes
 - ✓ Choice of branching variable: Select among variables in the relaxation problem whose optimal solution is not an integer.
 - ✓ Subdivision of subproblems: divide the value of a variable into two ranges, not just 0 and 1.
 - (ex) If the optimal solution gives a non-integer value x_j^* , then branch into
 - (1) $x_j \leq \lfloor x_j^* \rfloor$, (2) $x_j \geq \lfloor x_j^* \rfloor + 1$
 - ✓ Bounding: when determining the upper bound of the objective function (for maximization problems), rounding up or down is not allowed.
 - ✓ Fathoming test 3: check integer feasibility only for those variables subject to integer constraints in the relaxation solution.

The Branch-and-Bound Technique for MIP

- Branch-and-Bound Algorithm for MIP
 - Initialization: set $Z^* = -\infty$. Perform bounding, fathoming, and optimality test on the original problem.
 - ✓ If the original problem is not fathomed, proceed with iterations.
 - Iteration:
 - ✓ Branching: Select the most recently created subproblem among the remaining ones (if tied, choose the one with the higher upper bound)
 - From the integer-constrained variables, choose one that takes a non-integer value in the LP relaxation optimal solution.
 - Let this variable x_j , with LP relaxation solution value x_j^* .
 - Create two subproblems: (1) $x_j \leq \lfloor x_j^* \rfloor$ and (2) $x_j \geq \lfloor x_j^* \rfloor + 1$
 - ✓ Bounding: For each subproblem, solve the LP relaxation and derive its objective value Z as the bound.
 - ✓ Fathoming: For each subproblem, perform the following tests. If any are satisfied, the subproblem is fathomed:
 - 1. The subproblem's bound $\leq Z^*$ (for maximization problems)
 - 2. The LP relaxation of the subproblem is infeasible
 - 3. The LP relaxation optimal solution is integer-feasible with respect to all integer-constrained variables.
 - If this solution's objective value Z is better than Z^* , update Z^* . Then reapply test 1 to all remaining subproblems.
 - Optimality test: If no subproblem remain, terminate the algorithm (at this point, the best candidate solution Z^* is the optimal solution.)

The Branch-and-Bound Technique for MIP

- Branch-and-Bound Algorithm for MIP - Example
 - Maximize $Z = 4x_1 - 2x_2 + 7x_3 - x_4$
 - ✓ $x_1 + 5x_3 \leq 10$
 - ✓ $x_1 + x_2 - x_3 \leq 1$
 - ✓ $6x_1 - 5x_2 \leq 0$
 - ✓ $-x_1 + 2x_3 - 2x_4 \leq 3$
 - ✓ $x_j \geq 0$ for $j = 1, 2, 3, 4$
 - ✓ x_j is integer for $j = 1, 2, 3$ (i.e., $I = 3$)

The Branch-and-Bound Technique for MIP

- Branch-and-Bound Algorithm for MIP - Example

- Initialization.

- ✓ $Z^* = -\infty$

- ✓ LP relaxation optimum = $(5/4, 3/2, 7/4, 0)$ with $Z = 14.25$

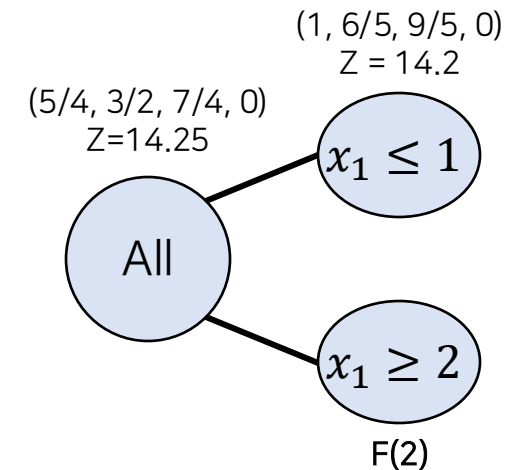
- Iteration 1.

- ✓ Subproblem 1: $x_1 \leq 1$

- LP relaxation optimum = $(1, 6/5, 9/5, 0)$ with $Z = 14.2$

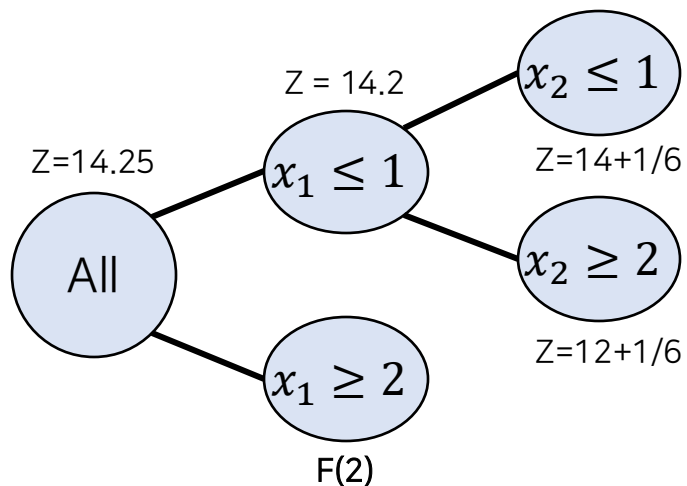
- ✓ Subproblem 2: $x_1 \geq 2$

- LP relaxation is infeasible.



The Branch-and-Bound Technique for MIP

- Branch-and-Bound Algorithm for MIP - Example
 - Iteration 2.
 - ✓ Subproblem 3: $x_1 \leq 1, x_2 \leq 1$
 - LP relaxation optimum = $(5/6, 1, 11/6, 0)$ with $Z = 14 + 1/6$
 - ✓ Subproblem 4: $x_1 \leq 1, x_2 \geq 2$
 - LP relaxation optimum = $(5/6, 2, 11/6, 0)$ with $Z = 12 + 1/6$

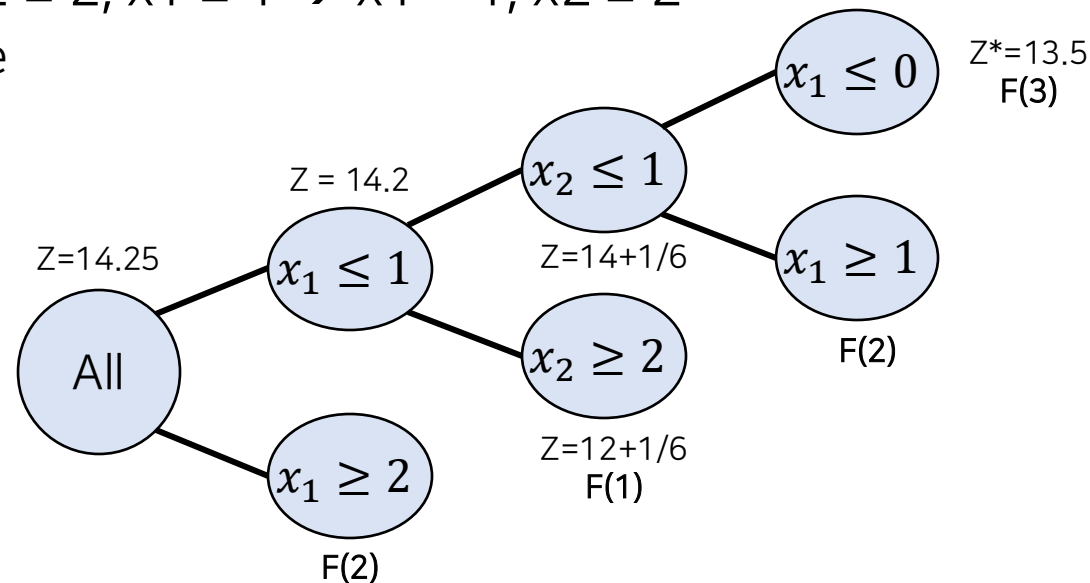


The Branch-and-Bound Technique for MIP

▪ Branch-and-Bound Algorithm for MIP - Example

• Iteration 3.

- ✓ Subproblem 5: $x_1 \leq 1, x_2 \leq 1, x_1 \leq 0 \rightarrow x_1 = 0, x_2 \leq 1$
 - LP relaxation optimum = $(0, 1, 2, 1/2)$ with $Z = 13.5$
 - Since all integer-constrained variables take integer values, fathom by test 3.
 - Also, update $Z^*=13.5$, and then fathom all other remaining subproblems by test 1.
- ✓ Subproblem 6: $x_1 \leq 1, x_2 \geq 2, x_1 \geq 1 \rightarrow x_1 = 1, x_2 \geq 2$
 - LP relaxation is infeasible



The Branch-and-Cut for BIP

- Background
 - Developed to avoid the explosive increase in computational complexity as problem size grows.
 - A key point is that it cannot be applied in all cases; a sparse constraint coefficient matrix is required.
- Automatic preprocessing for Pure BIP
 - Automatic problem preprocessing: Re-formulation aimed at solving the problem faster without excluding feasible solutions.
 - ✓ 1. Variable Fixing: Identify and eliminate values (0/1) that a binary variable cannot take in the optimal solution, thereby fixing the variable.
 - ✓ 2. Redundant constraint removal: Detect and eliminate redundant constraint.
 - ✓ 3. Tightening constraint: Tighten constraints so that the LP relaxation feasible region is reduced without excluding any feasible integer solutions.

The Branch-and-Cut for BIP

- Variable Fixing

- If one value of a variable cannot satisfy a given constraint in any way, then the variable is fixed to the other value.
- Example 1. For a \leq constraint, the largest positive coefficient plus the sum of all negative coefficients exceeds the RHS, then the variable with the largest positive coefficient is fixed to 0.
 - ✓ $3x_1 \leq 2 \rightarrow x_1 = 0$
 - ✓ $3x_1 + x_2 \leq 2 \rightarrow x_1 = 0$
 - ✓ $5x_1 + 3x_2 - 2x_3 \leq 2 \rightarrow x_1 = 0$

The Branch-and-Cut for BIP

- Variable Fixing

- If one value of a variable cannot satisfy a given constraint in any way, then the variable is fixed to the other value.
- Example 2. For a \geq constraint, if the sum of all positive coefficient variables except the one with the largest positive coefficient is smaller than RHS, then the variable with the largest positive coefficient is fixed to 1.
 - ✓ $3x_1 \geq 2 \rightarrow x_1 = 1$
 - ✓ $3x_1 + x_2 \geq 2 \rightarrow x_1 = 1$
 - ✓ $3x_1 + x_2 - 2x_3 \geq 2 \rightarrow x_1 = 1$

The Branch-and-Cut for BIP

- Variable Fixing

- If one value of a variable cannot satisfy a given constraint in any way, then the variable is fixed to the other value.
- Example 3. For a \geq constraint, if the sum of the smallest negative coefficient and all the other positive coefficient variables is less than the RHS, then the variable with the smallest negative coefficient is fixed to 0.
 - ✓ $x_1 + x_2 - 2x_3 \geq 1 \rightarrow x_3 = 0$

The Branch-and-Cut for BIP

- Variable Fixing

- If one value of a variable cannot satisfy a given constraint in any way, then the variable is fixed to the other value.
- Example 4. In a similar manner, multiple variables can be fixed within a single constraint.

$$\begin{aligned} \checkmark \quad 3x_1 + x_2 - 3x_3 \geq 2 &\rightarrow x_1 = 1 \text{ (since } 3 \times 0 + 1 \times 1 - 3 \times 0 < 2) \\ &\quad x_3 = 0 \text{ (since } 3 \times 1 + 1 \times 1 - 3 \times 1 < 2) \end{aligned}$$

- Example 5. In a similar manner, variable fixing is possible for constraints with a negative RHS.

$$\begin{aligned} \checkmark \quad 3x_1 - 2x_2 \leq -1 &\rightarrow x_1 = 0 \text{ (since } 3 \times 1 - 2 \times 1 > -1) \\ &\quad x_2 = 1 \text{ (since } 3 \times 0 - 2 \times 0 > -1) \end{aligned}$$

The Branch-and-Cut for BIP

- Variable Fixing

- If one value of a variable cannot satisfy a given constraint in any way, then the variable is fixed to the other value.
- Example 6. If a variable is fixed in one constraint, this can trigger a chain reaction where new variable fixings become possible in other constraints.
 - ✓ $3x_1 + x_2 - 2x_3 \geq 2 \rightarrow x_1 = 1$ (since $3 \times 0 + 1 \times 1 - 2 \times 0 < 2$)
 - ✓ $x_1 + x_4 + x_5 \leq 1 \rightarrow x_4 = 0 \text{ \& } x_5 = 0$ (since $x_1 = 1$)
 - ✓ $-x_5 + x_6 \leq 0 \rightarrow x_6 = 0$ (since $x_5 = 0$)

The Branch-and-Cut for BIP

- Variable Fixing

- If one value of a variable cannot satisfy a given constraint in any way, then the variable is fixed to the other value.
- Example 7. Sometimes, by combining one or more mutually exclusive alternative constraints, it becomes possible to fix variables in the newly derived constraint.
 - ✓ $8x_1 - 4x_2 - 5x_3 + 3x_4 \leq 2$
 - ✓ $x_2 + x_3 \leq 1$
 - ✓ $\rightarrow 8 \times 1 - \max\{4, 5\} \times 1 + 3 \times 0 > 2 \rightarrow x_1 = 0$

| The Branch-and-Cut for BIP

- Variable Fixing
 - In addition, there exist further variable fixing techniques that incorporate considerations of optimality.

The Branch-and-Cut for BIP

- Elimination of Redundant Constraints
 - If a functional constraint is still satisfied even when assigning the most unfavorable values, then the constraint is redundant with respect to the previous ones and can be removed.
 - For a \leq constraint, the most unfavorable binary assignment is: non-negative coefficient variables are 1, the rest (negative-coefficient variables) are 0.
 - For a \geq constraint, it is the opposite.
 - Example.
 - ✓ $3x_1 + 2x_2 \leq 6$
 - ✓ $3x_1 - 2x_2 \leq 3$
 - ✓ $3x_1 - 2x_2 \geq -3$
 - In many cases, once some variables are fixed, existing constraints may become redundant and can be removed.

The Branch-and-Cut for BIP

- Tightening Constraints
 - Example.
 - ✓ $\text{Max } Z = 3x_1 + 2x_2$
 - $2x_1 + 3x_2 \leq 4$
 - x_1, x_2 are binary
 - The functional constraint can be tightened to $x_1 + x_2 \leq 1$ without changing the feasible solution of the BIP. However, the feasible region of the LP relaxation problem is reduced.

The Branch-and-Cut for BIP

- Tightening Constraints

- Procedure for tightening $a \leq$ constraint

- ✓ $a_1x_1 + a_2x_2 + \dots + a_nx_n \leq b$
 - ✓ 1. S = the sum of all positive coefficients a_j .
 - ✓ 2. Identify any $a_j \neq 0$ such that $S < b + |a_j|$.
 - (a) If no such a_j exists, tightening not possible (stop)
 - (b) If $a_j > 0$, go to Step 3
 - (c) If $a_j < 0$, go to Step 4
 - ✓ 3. ($a_j > 0$) Compute $\bar{a}_j = S - b$, $\bar{b} = S - a_j$. Then, replace the constraint with $a_j = \bar{a}_j$, $b = \bar{b}$ and return to Step 1.
 - ✓ 4. ($a_j < 0$) Increase $a_j = b - S$, then return to Step 1.

The Branch-and-Cut for BIP

- Tightening Constraints

- Example

- ✓ Max $Z = 3x_1 + 2x_2$

- $2x_1 + 3x_2 \leq 4$

- x_1, x_2 are binary

- ✓ Iteration 1.

- (1) $S = 2 + 3 = 5$

- (2) Since $5 < 4 + 2$ ($S < b + |a_1|$) and $5 < 4 + 3$ ($S < b + |a_2|$), both conditions satisfied.

- (3) $\bar{a}_1 = 5 - 4 = 1, \bar{b} = 5 - 2 = 3 \rightarrow x_1 + 3x_2 \leq 3$

- ✓ Iteration 2.

- (1) $S = 1 + 3 = 4$

- (2) $4 < 3 + 3$ ($S < b + |a_2|$), condition satisfied for a_2

- (3) $\bar{a}_2 = 4 - 3 = 1, \bar{b} = 4 - 3 = 1 \rightarrow x_1 + x_2 \leq 1$

- ✓ Iteration 3.

- (1) $S = 1 + 1 = 2$, (2) For all $a_j \neq 0$, $S < b + |a_j|$ is not satisfied \rightarrow Stop.

- ✓ Final tightened constraint: $x_1 + x_2 \leq 1$

| The Branch-and-Cut for BIP

- Tightening Constraints
 - Additional Example.
 - ✓ $4x_1 - 3x_2 + x_3 + 2x_4 \leq 5$
 - $2x_1 - 3x_2 + x_3 + 2x_4 \leq 3$
 - $2x_1 - 2x_2 + x_3 + 2x_4 \leq 3$

The Branch-and-Cut for BIP

- Cutting Planes in Pure BIP
 - A cutting plane (or Cut) in IP is an additional constraint that reduces the feasible region of the LP relaxation without excluding any feasible integer solutions.
 - ✓ The tightening procedure we saw earlier is also a type of cut. Beyond that, many systematic cutting-plane generation methods have been developed.
 - Example.
 - ✓ For binary variables x_1, x_2, x_3 , $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$ can be tightened to $x_1 + x_2 + x_4 \leq 2$.
 - ✓ This new inequality (cut) eliminates fractional LP relaxation solutions that would otherwise satisfy the original inequality, while preserving all integer feasible solutions.

The Branch-and-Cut for BIP

- Cutting Planes in Pure BIP
 - Cutting plane generation procedure
 - ✓ 1. Consider a \leq constraint that has only non-negative coefficient.
 - ✓ 2. Search for an integer set (Minimum cover) that satisfies:
 - (a) If all variables in this set are assigned 1 and the others 0, the constraint is violated.
 - (b) If at least one variable in this set is set to 0, the constraint is satisfied.
 - ✓ 3. If the number of variables in this set is N , then the generated cutting plane is "Sum of all variables in the set $\leq N - 1$ "
 - Example.
 - ✓ From the previous example, $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$, the set $\{x_1, x_2, x_4\}$ is a Minimum cover. Thus, the cutting plane is " $x_1 + x_2 + x_4 \leq 2$ ".

Q&A