Chap11 Integer Programming

- ❑ **Sometimes, integer variables mean reality.**
- ❑ **LP + All variables must be integers = IP (Integer Programming)**
  - ✓ If integer variables are either 0 or 1 (binary variables) ➔ Binary IP
    - ➢ $x_j = 1$, if decision j is yes.
    - ➢ $x_j = 0$, if decision j is no.
  - ✓ If only some variables are required to be integers ➔ Mix IP
- ❑ **Prototype Example**
  - ✓ Build a factory in either City1 or City2, or both cities.
  - ✓ Build a new warehouse, but the location is restricted to where a new factory is being built.
  - ✓ Total capital available is 10 million.

Decision Number	Yes-or-No Question	Decision Variables	Net Present Value	Capital Required
1	Build factory in City1	x_1	9 (million)	6 (million)
2	Build factory in City2	x_2	5	3
3	Build warehouse in City1	x_3	6	5
4	Build warehouse in City2	x_4	4	2

❑ Innovative Uses of Binary Variables – Either-Or Constraints

- ✓ There are two constraints. At least one of them must hold. That is, only one (either one) must hold (whereas the other one can hold but is not required to do so).

  Either $3x_1 + 2x_2 \le 18$

  or $x_1 + 4x_2 \le 16$

- ✓ Let M be a very large positive number.
- ✓ Rewrite this requirement

  Either $3x_1 + 2x_2 \le 18$

  $x_1 + 4x_2 \le 16 + M$

  or $3x_1 + 2x_2 \le 18 + M$

  $x_1 + 4x_2 \le 16$

- ✓ The above is equivalent to the following

  $3x_1 + 2x_2 \le 18 + My$

  $x_1 + 4x_2 \le 16 + M(1 - y)$

  y is a binary variable (usually called the auxiliary variable).

❑ Innovative Uses of Binary Variables – *K* out of *N* Constraints Must Hold

- ✓ This is a direct generalization of the preceding case.

  $f_1(x_1, x_2, \ldots, x_n) \le d_1$

  $f_2(x_1, x_2, \ldots, x_n) \le d_2$

  ……

  $f_N(x_1, x_2, \ldots, x_n) \le d_N$

- ✓ $(N - K)$ of the constraints are not necessary need to be satisfied.

❑ Innovative Uses of Binary Variables – Functions with *N* Possible Values

$f_1(x_1, x_2, \ldots, x_n) = d_1$ or $d_2, \ldots,$ or d_n

- ✓ The equivalent IP

$$f_1(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{N} d_i y_i$$

$$\sum_{i=1}^{N} y_i = 1, \quad y_i \text{ is binary, for } i = 1, 2, \ldots, N$$

- ✓ An example
  - ➢ $3x_1 + 2x_2 = 6$ or 12 or 18.

❑ Innovative Uses of Binary Variables – The fixed-Charge Problem

$f(x) = k + cx$, if $x > 0$; $f(x) = 0$, if $x = 0$.

where x denotes the level of activity ($x \geq 0$), k denotes the setup cost, and c denotes the cost for each incremental unit.

- ✓ Whether taking the activity is represented by an auxiliary binary variable y.

$Z = ky + cx$

where $y = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \end{cases}$

- ✓ But, this is not LINEAR! So, adopt the technique used before to handle it.

$x \leq My$

- ✓ This will ensure that when $x > 0$, y must be 1.
- ✓ However, when $x = 0$, y is free to be either 0 or 1. But, we hope when $x = 0$, y is 0
- ✓ Luckily, the objective function takes care of this automatically.

❑ Innovative Uses of Binary Variables – Binary Representation of General Integer Variables

- ✓ In an IP problem, most of the variables are binary variables, only a few are general integer variables. (BIP can be solved more efficient).
- ✓ If a bounds on an integer variable x are $0 \le x \le u$ and if N is defined as the integer such that $2^N \le u \le 2^{N+1}$.
- ✓ Then, the binary representation of x is $x = \sum_{i=0}^{N} 2^i y_i$.

$x_1 \le 5$

$2x_1 + 3x_2 \le 30$

- ✓ The upper bound for x_1 is 5 and that of x_2 is 10. So, $N = 2$ for x_1 and $N = 3$ for x_2.

❑ More Examples – Making Choices When the Decision Variables Are Continuous

- ✓ Requirement 1: Three new products. At most two should be chosen to be produced.

- ✓ Requirement 2: Two plants. Just one is chosen to produce the new products and the production times are different.
- ✓ Let x_1, x_2, and x_3 are the production rates of the respective products.

		Product			Available Hours per Week
		1	**2**	**3**	
Plant	**1**	3	4	2	30
	2	4	6	2	40
Unit Profit		5	7	3	(thousands)
Sales Potential		7	5	9	(units per week)

- ✓ Initial formulation

- ✓ This initial formulation fails to reflect the real situation.
- ✓ To deal with requirement 1, we introduce three auxiliary binary variables (y_1, y_2, y_3). y_i = 1, if $x_j > 0$ (can produce product j). y_i = 0, if $x_j = 0$ (cannot produce product j).

- ✓ To deal with requirement 2, we introduce another auxiliary binary variable y_4. y_4 = 1, if Plant 2 is chosen ($4x_1 + 6x_2 + 2x_3 \leq 40$). y_4 = 0, if Plant 1 is chosen ($3x_1 + 4x_2 + 2x_3 \leq 30$).

- ✓ So, the complete model is

❑ More Examples – Violating Proportionality

- ✓ A company wants to buy 5 TV spots for promoting 3 products.
- ✓ The problem is how to allocate the 5 spots to these 3 products, with a maximum of three spots (and a minimum of zero) for each product.

Number of TV spots	**Profit**		
	Product		
	1	**2**	**3**
0	0	0	0
1	1	0	-1
2	3	2	2
3	3	3	4

- ✓ One formulation
  - ➢ Let x_1, x_2, and x_3 be the number of spots allocated to the respective products. However, this violates the proportionality property.
  - ➢ We introduce an auxiliary binary variable y_{ij} for each integer value of $x_i = j$ (j = 1, 2, 3).

- The resulting linear BIP model is

More Examples – Covering All Characteristics

- ✓ An airline needs to assign its crews to cover all its upcoming flights.
- ✓ Only three of the crew based will be assigned.

Flights	**Feasible Sequence of Flights**											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
S.F. to L.A.	1			1			1			1		
S.F. to Denver		1			1			1			1	
S.F. to Seattle			1			1			1			1
L.A. to Chicago				2			2		3	2		3
L.A. to S.F.	2					3				5	5	
Chicago to Denver				3	3				4			
Chicago to Seattle							3	3		3	3	4
Denver to S.F.		2		4	4				5			
Denver to Chicago					2			2			2	
Seattle to S.F.			2				4	4				5
Seattle to L.A.						2			2	4	4	2
Cost, $1,000’s	2	3	4	6	7	5	7	8	9	9	8	9

❑ Is IP easy to solve?

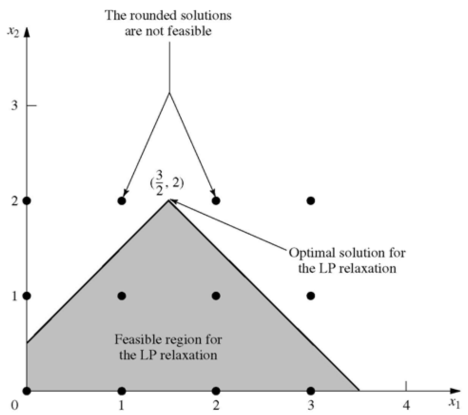
- ✓ At the first glance, IP may be easy to solve since the number of feasible solution is smaller and is limited. But…
- ✓ If the feasible region is bounded, the number of feasible solution is finite. But, the number is simply too large (exponential growth). With n variables, there are 2^n solutions to be considered.
- ✓ The corner point is (generally) no longer feasible.

❑ LP relaxation—Take out the integer requirement from an IP problem.

- ✓ Which will have better objective value? IP or its LP relaxation?

❑ Why can't we solve the LP relaxation and then round it up/down to get the required integer solution?

- ✓ Rounding does not guarantee the feasibility.



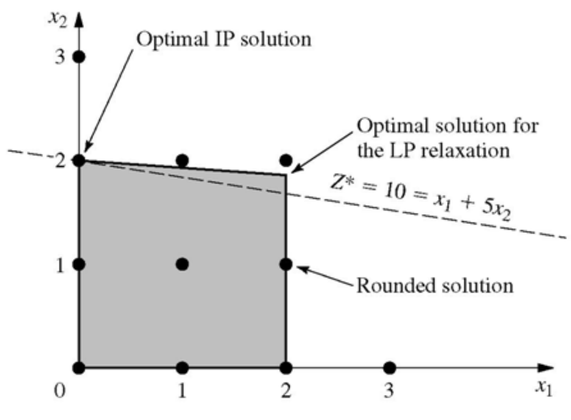
Max $Z = x_2$

S.T. $-x_1 + x_2 \le 0.5$

$x_1 + x_2 \le 3.5$

$x_1, x_2 \ge 0$ and are integers

- ✓ Rounding does not guarantee the optimality.



Max $Z = x_1 + 5x_2$

S.T. $x_1 + 10x_2 \le 20$

$x_1 \le 2$

$x_1, x_2 \ge 0$ and are integers

❑ The Branch-and-Bound Technique and Its Application to Binary IP

- ✓ Basic Idea—Enumeration procedure can always find the optimal solution for any bounded IP problem. But it takes too much time. So, we consider the partial enumeration. That is, divide and conquer.

- ✓ Bounding, branching, and fathoming are the three components of Branch-and-Bound technique

$$
\begin{array}{llcl}
\text{Max } Z = 9x_1 + 5x_2 + 6x_3 + 4x_4 & & & \\
\text{S.T.} & 6x_1 + 3x_2 + 5x_3 + 2x_4 & \leq & 10 \\
& x_3 + x_4 & \leq & 1 \\
& -x_1 + x_3 & \leq & 0 \\
& -x_2 + x_4 & \leq & 0 \\
\end{array}
$$

x_j is binary, j = 1,2,3,4

❑ Bounding—For each problem or subproblem (will define later), we need to obtain a bound on how good its best feasible solution can be.

- ✓ Usually, the bound is obtained by solving the **LP relaxation**.

- ✓ The solution for the LP relaxation is: $(x_1, x_2, x_3, x_4) = (5/6, 1, 0, 1)$, $Z = 16.5$.

- ✓ If LP relaxation does not yield an integer solution, the optimal value (of LP relaxation) is a bound (lower bound, if it is a Min problem; upper bound, if it is a Max problem) of the original IP problem.

- ✓ That is $Z^* \leq 16.5$ or 16 (round off because of integer property).

❑ Branching—This is the "divide" part of Branch-and-Bound. Generally, we fix the value of one of the variables.

- ✓ Subproblem 1: Fix $x_1 = 0$.

$$
\begin{array}{llc}
\text{Max } Z = 5x_2 + 6x_3 + 4x_4 & & \\
\text{S.T.} & 3x_2 + 5x_3 + 2x_4 & \leq \\
& x_3 + x_4 & \leq \\
& x_3 & \leq \\
& -x_2 + x_4 & \leq \\
\end{array}
$$

x_j is binary, j = 2, 3, 4

✓ Subproblem 2: Fix $x_1 = 1$.

$$\begin{array}{llr}
\text{Max } Z = ____ + 5x_2 + 6x_3 + 4x_4 & & \\
\text{S.T.} & 3x_2 + 5x_3 + 2x_4 & \leq \\
& x_3 + x_4 & \leq \\
& x_3 & \leq \\
& -x_2 \qquad + x_4 & \leq
\end{array}$$

x_j is binary, $j = 2, 3, 4$

- **For each subproblem, continue with the bounding procedures.**
  - ✓ Continue to obtain bounds for the two subprobems in the same way.
    - ➢ For subproblem 1: (fixing $x_1 = 0$) Solve its LP relaxation.

      $(x_1, x_2, x_3, x_4) = (0, 1, 0, 1)$, $Z = 9$. It is an integer solution. That is, incumbent (the best feasible solution found so far) = 9.
    - ➢ For subproblem 2: (fixing $x_1 = 1$) Solve its LP relaxation.

      $(x_1, x_2, x_3, x_4) = (1, 4/5, 0, 4/5)$, $Z = 16.5$. If we follow this branch, the best we can get is 16.

- **Fathoming—A subproblem can be conquered (fathomed), and thereby dismissed from further consideration.**
  - ✓ The subproblem has integer solutions. There is no need to branch on this subproblem anymore. Set the incumbent solution to this solution, if it is better than the current incumbent.
    - ➢ Thus, subproblem 1 is fathomed.
  - ✓ There is no reason to consider further any subproblem whose bound $\leq$ the incumbent solution. (for a max. problem).
    - ➢ Fathomed on any subproblem whose bound $\leq$ 9 in our example.

- ➢ This outcome does not occur in the current iteration of our example because the bound for subproblem 2 is 16 in our example. We still have chance to find a better solution than the incumbent.

- ✓ If the subproblem has no feasible solution, it is fathomed.

❑ Summary of Fathoming Tests (for a max problem)

- ✓ Test 1: Its bound $\leq$ incumbent.

- ✓ Test 2: Its LP relaxation has no feasible solution.

- ✓ Test 3: The optimal solution for its LP relaxation is integer. If this solution is better than the incumbent, it becomes the new incumbent, and Test 1 is applied to all unfathomed subproblems with the new incumbent.

❑ Summary of the Binary Branch-and-Bound Algorithm (max problem)

- ✓ Initialization: Set incumbent = negative infinite. Solve the LP relaxation. If obtain integer solution, it is done and Z is the optimal solution ($Z^* = Z$). Otherwise, implement the iteration below.

- ✓ Steps for each iteration:

  - ➢ Branching: Among the remaining (unfathomed) subproblems, select the one that was created most recently. Break ties according to which has the larger bound. Branch from the node for this subproblem to create two new subproblems by fixing the next variable at either 0 or 1 (follow the order of x_1, x_2, x_3, ….) .

  - ➢ Bounding: For each new subproblem, obtain its bound by solving its LP relaxation and rounding down the value of Z.

  - ➢ Fathoming: For each new subproblem, apply the three fathoming tests, and discard those subproblems that are fathomed by any of the tests.

- ✓ Optimality Test: Stop when there are no remaining unfathomed subproblems; the incumbent is the optimal. Otherwise, perform another iteration.

❑ Completing the Example

✓ Iteration 2: Branch on x_2.

➢ Subproblem 3 ($x_1 = 1$, $x_2 = 0$)

$$
\begin{aligned}
\text{Max } Z = \_\_\_\_ &+ 6x_3 + 4x_4 \\
\text{S.T.} \quad & 5x_3 + 2x_4 \leq \\
& x_3 + x_4 \leq \\
& x_3 \leq \\
& + x_4 \leq
\end{aligned}
$$

x_j is binary, $j = 3, 4$

➢ Solve its LP relaxation. We have $(x_1, x_2, x_3, x_4) = (1, 0, 4/5, 0)$, $Z = 13.8$. If we follow this branch, the best we can get is 13.

➢ Subproblem 4 ($x_1 = 1$, $x_2 = 1$)

$$
\begin{aligned}
\text{Max } Z = \_\_\_\_ &+ 6x_3 + 4x_4 \\
\text{S.T.} \quad & 5x_3 + 2x_4 \leq \\
& x_3 + x_4 \leq \\
& x_3 \leq \\
& + x_4 \leq
\end{aligned}
$$

x_j is binary, $j = 3, 4$

➢ Solve its LP relaxation. We have $(x_1, x_2, x_3, x_4) = (1, 1, 0, 1/2)$, $Z = 16$. If we follow this branch, the best we can get is 16.

➢ Cannot fathom any subproblem.

✓ Iteration 3: Branch on x_3.

➢ Subproblem 5 ($x_1 = 1$, $x_2 = 1$, $x_3 = 0$) : Solve its LP relaxation. We have $(x_1, x_2, x_3, x_4) = (1, 1, 0, 1/2)$, $Z = 16$. If we follow this branch, the best we can get is 16.

➢ Subproblem 6 ($x_1 = 1$, $x_2 = 1$, $x_3 = 1$) : Solve its LP relaxation. It is infeasible. Fathomed.

✓ Iteration 4: Branch on x_4.

➢ Subproblem 7 ($x_1 = 1$, $x_2 = 1$, $x_3 = 0$, $x_4 = 0$): Solve (not really necessary) its LP relaxation. We have an integer feasible solution $(x_1, x_2, x_3, x_4) = (1, 1, 0, 0)$ with $Z = 14$. This is better than the current incumbent (9). Set the new incumbent = 14.

➢ Check all other unfathomed subproblem, we can fathom subproblem 3.

- Subproblem 8 ($x_1 = 1$, $x_2 = 1$, $x_3 = 0$, $x_4 = 1$): It is infeasible. Fathomed.
- No unfathomed subproblem left, we are done.
- The optimal solution is $x_1^* = 1$, $x_2^* = 1$, $x_3^* = 0$, $x_4^* = 0$, $Z^* = 14$.

- **What will we do if it is a Min problem?**
  - ✓ Convert to Max problem.
  - ✓ Change the rule of "fathom by bound".

❑ A Branch-and-Bound Algorithm for Mixed Integer Programming

- ✓ The choice of the branching variable: only consider integer-restricted variables that have a noninteger value in the optimal solution for the LP relaxation.
- ✓ How to fix the chosen variable: $x_j \le [x_j^*]$ (greatest integer $\le x_j^*$), $x_j \ge [x_j^*]+1$
  - ➢ For example: $x_2 = 3.5 \rightarrow x_2 \le 3$ and $x_2 \ge 4$.
- ✓ Do not round down when the bound is set due to the mixed IP nature.
- ✓ Fathoming Test 3 (fathom by solving) only considers the integer-restricted variables.

❑ A Mixed IP Example

$$
\begin{aligned}
\text{Max } Z = {} & 4x_1 - 2x_2 + 7x_3 - x_4 \\
\text{S.T.} \quad & x_1 + 5x_3 \le 10 \\
& x_1 + x_2 - x_3 \le 1 \\
& 6x_1 - 5x_2 \le 0 \\
& -x_1 + 2x_3 - 2x_4 \le 3 \\
& x_j \ge 0, \ j = 1, 2, 3, 4 \\
& x_j \text{ is an integer, } j = 1, 2, 3
\end{aligned}
$$

- **How to improve the efficiency of Branch-and-Bound for pure BIP? -- Fixing Variables.**
  - ✓ If one value of a variable cannot satisfy a certain constraint, then that variable should be fixed at its other value.

$3x_1 \le 2 \rightarrow x_1 = 0$

$3x_1 + x_2 <= 2 \rightarrow x_1 = 0$

$5x_1 + x_2 - 2x_3 \le 2 \rightarrow x_1 = 0$

  - ✓ General rule for checking <= constraints: Identify the variable with the largest positive coefficient, and if the sum of that coefficient and any negative coefficients exceeds the right-hand side, then that variable should be fixed at 0.
  - ✓ More Examples for >= constraints (using similar reasoning).

$3x_1 \ge 2 \rightarrow x_1 = 1$
$3x_1 + x_2 \ge 2 \rightarrow x_1 = 1$
$5x_1 + x_2 - 2x_3 \ge 2 \rightarrow x_1 = 1$
$x_1 + x_2 - 2x_3 \ge 1 \rightarrow x_3 = 0$
$3x_1 + x_2 - 3x_3 \ge 2 \rightarrow x_1 = 1$ and $x_3 = 0$
$3x_1 - 2x_2 \le -1 \rightarrow x_1 = 0$ and $x_2 = 1$

  - ✓ Fixing a variable from one constraint can sometimes generate a chain reaction of then being able to fix other variables from other constraints.

$3x_1 + x_2 - 2x_3 \ge 2 \rightarrow x_1 = 1$, Then
$x_1 + x_4 + x_5 <= 1 \rightarrow x_4 = 0$ and $x_5 = 0$, Then
$-x_5 + x_6 \le 0 \rightarrow x_6 = 0$.

  - ✓ Sometimes, it is possible to combine one or more mutually exclusive alternatives constraints with another constraint to fix a variable.

$8x_1 - 4x_2 - 5x_3 + 3x_4 \le 2$
$x_2 + x_3 \le 1$
$\rightarrow x_1 = 0$.

- **How to improve the efficiency of Branch-and-Bound for pure BIP? -- Eliminating Redundant Constraints.**

$3x_1 + 2x_2 \le 6$
$3x_1 - 2x_2 \le 3$
$3x_1 - 2x_2 \ge -3$

  - ✓ This case usually happens when we fix some variables.

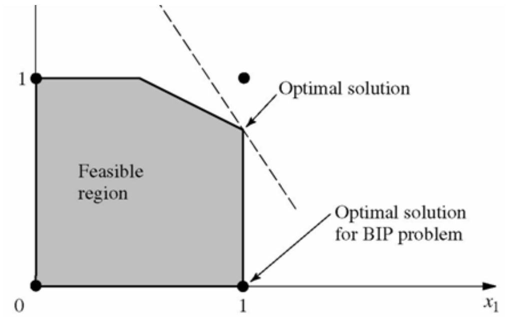
❑ How to improve the efficiency of Branch-and-Bound for pure BIP? -- Tightening constraints

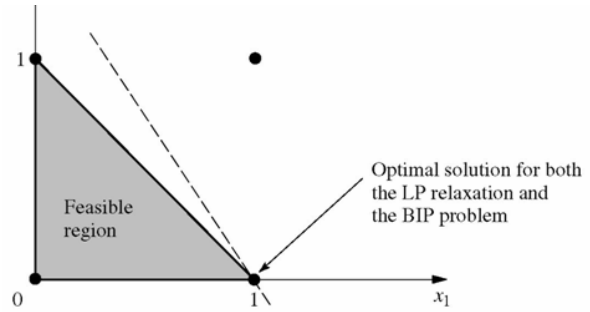
Max $Z = 3x_1 + 2x_2$

S.T. $2x_1 + 3x_2 \le 4$

x_1, x_2 binary

✓ If replace the first constraint by $x_1 + x_2 \le 1$, the optimum of LP relaxation is the optimum of Binary IP. Notice that we do not cut off any feasible solution.





✓ Rationale of tightening a $\le$ for BIP problems.

✓ Procedure for tightening a $\le$ constraint for BIP problems.

- (Step 1) Calculate S = sum of the positive a_j.
- (Step 2) Identify an a_j != 0 such that $S < b + |a_j|$.

  (a) If none, stop; cannot tighten any more.

  (b) If $a_j > 0$, go to step 3

  (c) If $a_j < 0$, go to step 4.
- (Step 3: $a_j > 0$) Calculate $a_j' = \underline{S} - b$ and $b' = S - a_j$. Reset $a_j = a_j'$ and $b = b'$. Return to step 1.
- (Step 4: $a_j < 0$) Increase a_j to $a_j = b - S$. Return to step 1.

❑ Generating Cutting Planes for Pure BIP

- ✓ A cut plane (cut) is a new constraint that reduces the feasible region for the LP relaxation without eliminating any feasible solutions for the original IP problem.

$$
\begin{aligned}
\text{Max } Z &= 9x_1 + 5x_2 + 6x_3 + 4x_4 \\
\text{S.T. } & 6x_1 + 3x_2 + 5x_3 + 2x_4 \le 10 \\
& x_3 + x_4 \le 1 \\
& -x_1 + x_3 \le 0 \\
& -x_2 + x_4 \le 0 \\
& x_1 \sim x_4 \text{ binary}
\end{aligned}
$$

- ✓ The binary constraints and constraint $6x_1 + 3x_2 + 5x_3 + 2x_4 \le 10$ together imply that $x_1 + x_2 + x_4 \le 2$.

- ✓ This cut eliminates part of the feasible region (including optimal), but does not eliminate any feasible solution for IP problem.

- ✓ Adding this constraint may improve the efficiency of Branch-and-Bound.
  - ➢ More nodes may be fathomed by solving or by bounding.

- ✓ General procedure for finding a cut (out of several common used methods).
  - Consider any constraint in $\leq$ form with only nonnegative coefficients.
  - Find a group of variables (called **minimum cover**) such that

    (a) The constraint is violated if every variable in the group equals 1 and all other variables equal 0.

    (b) The constraint becomes satisfied if the value of any one of these variables is changed from 1 to 0.
  - By letting N denote the number of variables in the group, the resulting cutting plane has the form

    Sum of variables in group $\leq N - 1$.