# Integer Programming

Renda Nyamande

30 August 2022

**Abstract**

**We shall conquer at all costs!**

# What is it?

**Def** LP problems with at least one variable that has to be an integer.

## Types

- Pure - All decision vars have to be integers. $X \in int$
- Mixed - Some ints and some not (Even int + binary).
- Binary - Decision vars can only take on the values 0 or 1. $X \in \{0; 1\}$

## Characteristics

- The change from a normal LP to an integer one changes the problem drastically; Making it more complex (Might not even have a solution).

## Example 1

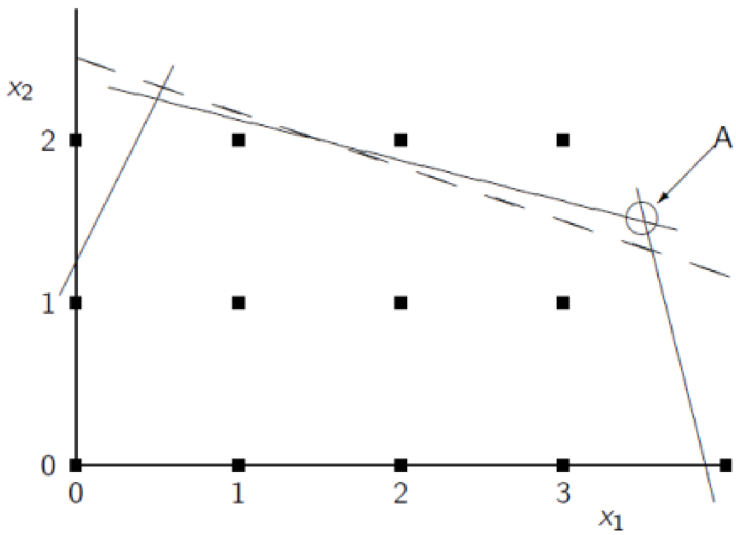


Figure 1: Example 1

Max $X_1 + 3X_2$
$S.t:$
$2X_1 + 8X_2 \leq 9$
$4X_1 + X_2 \leq 15.5$
$8X_1 - 4X_2 \geq -5$
$X_1, X_2 \geq 0$ and Integer

- Point A would be the optimal solution point if this was a normal LP problem.
- Trying to use the same point rounded up would result in one using a point that's actually not in the feasible region.
- If you round down, the solution would be feasible but not optimal.
- Adding an int constraint can only restrict the solution.
- Therefore A is the best solution we could ever hope to achieve having restricted things.
- So shifting the isocost line down from A (Restricting solution), the first int point it touches will be the optimal point.
- Theres an algorithm we'll use to find these solutions.

## Example 2 - Project selection problem

Suppose we have identified four investment opportunities and we have R14000 to invest. Project 1 requires an investment of R5,00 and has a present value (a time-discounted value) of R8,000; Project 2 requires R7,000 and has a value of R11,000; Project 3 requires R4,000 and has a value of R6,000; and Project 4 requires R3,000 and has a value of R4,000. The question is: into which projects should we place our money in order to maximize our total present value?

Figure 2: Project selection problem

$$Let X_j = \begin{cases} 1 & \text{if project } j \text{ is selected; j = 1,2,3,4} \\ 0 & \text{Otherwise} \end{cases} \tag{1}$$

Max $8000X_1 + 11000X_2 + 6000X_3 + 4000X4$
St:
$5000X_1 + 7000X_2 + 4000X_3 + 3000X_4 \leq 14000$
$X_1, X_2, X_3, X_4 \in \{0, 1\}$

- Using simplex, Z = R22000 ($X_1 = X_2 = 1, X_3 = 0.5, X_4 = 0$)
- This obviously doesn't satisfy the last constraint.
- If you round down, Z = R19000
- Better solution - $X_1 = 0, X_2 = X_3 = X_4 = 1$
- This is found using the algorithm we'll learn later.
- We could have added constraints:
  - Only 2 projects can be selected ($X_1 + X_2 + X_3 + X_4 = 2$).
  - If project 1 is selected, then project 3 must also be selected ($X_1 \leq X_3$)
  - If project 2 is selected, then project 4 can't be selected ($X_2 + X_4 \leq 1$) - He acknowledged the fact that this condition goes the other way around as well.

## Example 3 - Set covering problem

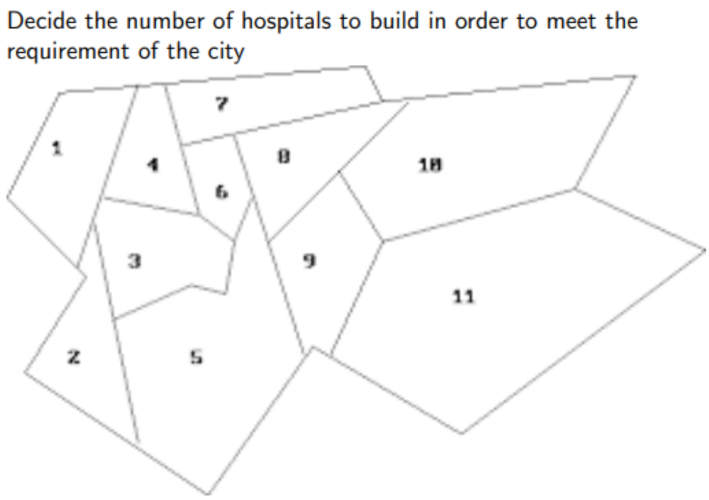


Figure 3: Set covering problem

Find min number of hospitals that can be built. Trying to give each city access to a hospital(They have to be adjacent to simulate access).

$$Let X_j = \begin{cases} 1 & \text{if hospital built in } j; j = \text{1,2,3..., 11} \\ 0 & \text{Otherwise} \end{cases} \quad (2)$$

Min $X_1 + X_2 + ... + X_{11}$
St:
$X_1 + X_2 + X_3 + X_4 \geq 1(X1)$
$X_1 + X_2 + X_3 + X_5 \geq 1(X2)$
$X_1 + X_2 + X_3 + X_5 + X_6 \geq 1(X3)$
$X_1 + X_3 + X_6 + X_7 \geq 1(X4)$
$X_2 + X_3 + X_6 + X_8 + X_9 \geq 1(X5)$
$X_3 + X_4 + X_5 + X_7 + X_8 + X_9 \geq 1(X6)$
$X_4 + X_6 + X_8 \geq 1(X7)$
$X_5 + X_6 + X_7 + X_9 + X_{10} \geq 1(X8)$
$X_5 + X_6 + X_8 + X_{10} + X_{11} \geq 1(X9)$
$X_8 + X_9 + X_{11} \geq 1(X10)$
$X_9 + X_{10} \geq 1(X11)$
$X_1, X_2, X_3, ...X_{11} \in \{0, 1\}$

- To get the constraints, we go from $X_1 to X_{11}$ - For each X, the sum of it and all adjacent (touching) X's must at least 1.
- This means we build at least one hospital to accomodate that group of cities. This will eventually cover all combinations of X's thus we'll be sure we ran through every possible hospital location (aka covered all cities).
- Optimal solution - $X_3 = X_8 = X_9 = 1$

## Example 4 - Fixed charge problem

| Operation | Prod 1 | Prod 2 | Prod 3 | Hours available |
|---|---|---|---|---|
| Machining | 2 | 3 | 6 | 600 |
| Griding | 6 | 3 | 4 | 300 |
| Assembly | 5 | 6 | 2 | 400 |
| Unit profit | R48 | R55 | R50 | |
| Setup cost | 1000 | 800 | 900 | |

Figure 4: Fixed charge problem

Let $X_j$ be the number of product j produced.

$$LetY_j = \begin{cases} 1 & \text{if product is produced; } j = 1,2,3 \\ 0 & \text{Otherwise} \end{cases} \tag{3}$$

Max $48X_1 + 55X_2 + 50X_3 - 1000Y_1 - 800Y_2 - 900Y_3$
St:
$2X_1 + 3X_2 + 6X_3 \leq 600$
$6X_1 + 3X_2 + 4X_3 \leq 300$
$5X_1 + 6X_2 + 2X_3 \leq 400$
$X_1 \leq M_1Y_1$
$X_2 \leq M_2Y_2$
$X_3 \leq M_3Y_3$
$Y_1, Y_2, Y_3 \in \{0; 1\}$
$X_1, X_2, X_3 \geq 0$ and Integer

- Manufacturing where you're making paint
- **Setup cost (fixed charge)** - Cost of cleaning machine after making red paint tp prepare to make yellow for example.
- **Variable cost** - e.g Hiring trucks will cost a certain, fixed, amount (fixed cost) and the cost of using them will vary (hence variable cost).
- The 4th to 6th constraints are added to make sure that each cost $Y_j$ is only applied if $X_j$ is actually produced.
  - M is a big number greater than 0.
  - if X is 0, Y can be 0 or 1.
  - BUT if X is 1, Y **HAS** to be 1 as well (Since MY has to be at least equal to X).
  - This is to say, surely if we produce at least one (X), there has to be some cost involved (Y).
  - In fact, the moment X is slightly above 0, Y is 1.
  - M is acting as an upper bound for X.
  - In this case, We can find a reasonable value for M1:
    * Method - For each X, in the first 3 constraints, if the other X values are 0, M will be the max value X can take:
      · For X1,
      · For constraint 1: X1 = 300

- For constraint 2: X1 = 50
- For constraint 3: X1 = 80
- Therefore, the max it can be will be the min of the 3 values (**We do this so that all constraints are satisfied**; Therefore, M1=50).

## Relaxation of ILP

- Original ILP

$$\max 2x_1 + 3x_2$$
$$\text{subject to :}$$
$$4x_1 + 12x_2 \leq 33$$
$$10x_1 + 4x_2 \leq 35$$
$$x_1, x_2 \geq 0$$
$$x_1, x_2 \quad \text{must be integers}$$

- LP Relaxation, drop integer constraints:

$$\max 2x_1 + 3x_2$$
$$\text{subject to :}$$
$$4x_1 + 12x_2 \leq 33$$
$$10x_1 + 4x_2 \leq 35$$
$$x_1, x_2 \geq 0$$

Figure 5: Relaxation of ILP

- When you drop the integer constraints, you relax the integer constraint, hence LP relaxation.
- If the solution we find just happens to be an integer solution, we're done, else, we have to do something else.
- The feasible region of the relaxed problem is a **superset** (contains) of the associated integer linear programming solution feasible region. This makes sense as restricting things can only make the feasible region smaller.
- The objective func value for the integer solution can't be greater than that of the relaxed (normal simplex) version in a maximization.

- The optimal solution of the relaxed solution is the upper bound of the ILP solution for a maximization.
- The optimal solution of the relaxed solution is the lower bound of the ILP solution for a minimization.
- At each iteration in our search for a solution, the objective function value for any feasible solution ILP is a lower bound for an maximixation (We can't accept anything lower since we already have at least that value as a potential solution).
- At each iteration in our search for a solution, the objective function value for any feasible solution ILP is an upper bound for an minimization (We can't accept anything above since we already have at most that value as a potential solution).
- This is the **Branch and Bound** algorithm.
- It involves solving a series of LP problems.
- It can solve any ILP; In fact, any optimization (It's just that some take forever).
- In practice, the branch and bound can take a lot of computational effort and time.