Lesson 8: Logical If/Then and Either/Or Constraints

1 A Binary IP

USNA is considering purchasing some new generators to ensure power in the case of storms. The table below gives the fixed cost and MW of power generated by each potential generator.

	Fixed cost ($ million)	Power Generated (MW)
Generator 1	1.20	3,000
Generator 2	0.75	4,000
Generator 3	0.50	5,000
Generator 4	1.00	3,500
Generator 5	1.10	3,800

USNA wants to ensure that at least 8500 MW of power are available if there is an outage.

1. Formulate a concrete IP that would tell USNA the optimal generators to purchase while satisfying power demands at minimum cost.

Variables which generators does USNA buy

let $x_i = 1$ if USNA buys generator i or 0 otherwise for $i \in \{1,2,3,4,5\}$

Objective

min cost: $1.2x_1 + 0.75x_2 + 0.5x_3 + 1x_4 + 1.1x_5$

Constraints

$x_i \in \{0,1\} \quad \forall\ i \in \{1,2,3,4,5\}$

$3000x_1 + 4000x_2 + 5000x_3 + 3500x_4 + 3800x_5 \geq 8500$

2 Types of Binary Variable Constraints

In general, there are several types of binary constraints we'd like to enforce:

- Fixed Charge (Lesson 6) → Weak and Strong forcing
- Mutually exclusive and multiple choice constraints
- If/then constraints
- Either/or constraints

] Hard / tricky constraints

3 Mutually Exclusive and Multiple Choice Constraints

The two simplest type of logical constraints deal with mutual exclusion or multiple choice.

- **Mutual Exclusive:** You are only allowed to choose a specific subset of variables to be 1. → Can't choose every variable
  - Suppose in the generator problem above, you are only allowed to choose at most 1 of generators 2, 3, and 4. Write a logical constraint that enforces this.

$$\underbrace{x_2 + x_3 + x_4}_{\text{1 if generators 2, 3, or 4 are chosen}} \underbrace{\le 1}_{\text{can choose at most 1 of these 3}}$$

- **Multiple Choice:** You must choose a certain subset of variables to be equal to 1.
  - Suppose that USNA knows that facilities will only approve the purchase of exactly 2 of generators 1, 2, 3, 4, and 5. Write a logical constraint that enforces this.

$$\underbrace{x_1 + x_2 + x_3 + x_4 + x_5}_{i} \underbrace{= 2}_{\text{exactly 2 are allowed to be chosen}}$$

$x_i = 1$ if choose a generator i

4 Introducing If/Then Constraints on Two Variables

Often, it is the case that we want to enforce if/then logic on binary variables. For example, perhaps Generator 1 and 3 are made by competing companies, so if Generator 1 is purchased, we can not purchase Generator 3.

If $x_1=1$ then $x_3=0$

2. Try to write a constraint to capture this logic. That is write an inequality which says if $x_1 = 1$ then $x_3 = 0$.

$x_1+x_3=1$

$x_1=1 \Rightarrow x_3=0$

$x_1=0 \Rightarrow x_3=1$

x_3 is not free (0 or 1) ✗

$x_1 \cdot x_3 = 0$

$x_1=1 \rightarrow x_3=0$

$x_1=0 \rightarrow x_3$ free

Satisfies constraint but nonlinear ✗

$x_1+x_3 \leq 1$

$x_1=1 \rightarrow x_3=0$

$x_1=0 \rightarrow x_3 \leq 1 \rightarrow x_3$ free

Works ✓

It can seem like a lot of guess and check in order to get a constraint that actually works. Fortunately, there's a process you can follow to make this significantly easier. Namely, the following steps can make writing these logical constraints much easier.

- Write the constraint as a conditional statement and convert all clauses to 1s (why?)

All clauses $=1 \Rightarrow$ If $x_1=1$ then $x_3=0 \rightarrow$ If $x_1=1$ then $(1-x_3)=1$

$x_3=0 \rightarrow 1-x_3=1$

- Write the constraint from left to right.
  - The left side of the **if** statement is the LHS of the constraint
  - The inequality is always $\leq$
  - If it is converted to 1s, everything following **then** is the RHS of the constraint. That is the high value on the left "pushes" the RHS to its high value.

$x_1 \leq 1-x_3$

LHS at its high value pushing the RHS to its high value

- **ALWAYS** check every logical constraint for **explicit** and **implicit** satisfaction.
  - **Explicit** satisfaction: Does it satisfy the written constraint?

    $x_1=1 \rightarrow 1 \leq 1-x_3 \rightarrow x_3=0$

  - **Implicit** satisfaction: If the constraint is not true is the RHS free?

    $x_1=0 \rightarrow 0 \leq 1-x_3$ $\nearrow x_3=0$, $\searrow x_3=1$] x_3 is free

Using the steps above, capture the following if/then constraint logic:

3. If we purchase generator 2 then we must purchase generator 4.

① if $x_2=1$ then $x_4=1$

$\underbrace{x_2=1}_{\text{LHS of constraint}}$ $\underbrace{\text{then}}_{\le}$ $\underbrace{x_4=1}_{\text{RHS}}$

② $\boxed{x_2 \le x_4}$

③ Explicit Satisfaction

$x_2=1 \rightarrow 1 \le x_4 \rightarrow x_4=1$ ✓

④ Implicit

$x_2=0 \rightarrow 0 \le x_4$

x_4 free

4. If we don't purchase generator 1 then we must purchase generator 5.

① If $x_1=0$ then $x_5=1$

Convert to 1s

$x_1=0 \rightarrow 1-x_1=1$

② $\boxed{1-x_1 \le x_5}$

③ $x_1=0 \rightarrow 1 \le x_5 \rightarrow x_5=1$ ✓

④ $x_1=1 \rightarrow 0 \le x_5$

x_5 free ✓

Works for All 2 Variable if/then

5. If we don't purchase generator 3 then we can't purchase generator 1. and we must buy generator 2

① If $x_3=0$ then $x_1=0$ and $x_2=1$

If $1-x_3=1$ then $1-x_1=1$ and $x_2=1$

② $(1-x_3) \le (1-x_1) + x_2$ → $\boxed{\begin{array}{l} 1-x_3 \le 1-x_1 \\ 1-x_3 \le x_2 \end{array}}$ ✓

$\boxed{2(1-x_3) \le (1-x_1) + x_2}$

③ $x_3=0$ $\quad 1 \le (1-x_1)+x_2$ ✗

$x_3=0 \rightarrow x_1=0$, $x_2=1$ ✓

$x_3=0$

$2 \le (1-x_1)+x_2$ $\quad x_1=0$, $x_2=1$ ✓

④ $x_3=1 \rightarrow 0 \le 1-x_1$ ✓

$\rightarrow 0 \le x_2$

$x_3=1$

$0 \le (1-x_1)+x_2$

x_1, x_2 free ✓

The same idea will also work with 3 or more variables, just the LHS or RHS of the constraints may be off by a constant factor. That's why it's important to always check explicit and implicity satisfaction.

$1 \le (1-x_1) + x_2$ $\quad x_1=0$, $x_2=1$ works but $x_1=0$, $x_2=0$ works

5 Either/Or Constraints: Motivating Example

The remainder of this lesson will focus on either/or constraints.

Quality Cabinets used an integer-program to determine how many of each type of cabinet to make in order to maximize profit. They used decision variables x_s, x_d, and x_e to represent the number of standard, deluxe, and enhanced cabinets, respectively, to produce each week. Each cabinet requires a certain number of hours of painting time and there is a limit on the number of painting hours available. A small part of the model is:

$$\begin{array}{ll} \text{Maximize} & 25x_a + 45x_d + 60x_e \\ \text{subject to} & 2x_a + 4x_d + 5x_e \leq 700 \qquad \text{(painting time)} \end{array}$$

Xa → Standard
Xe → enhanced
Xd → deluxe

$\underline{2}xa + \underline{4}xd + \underline{5}xe \leq 700$

painting time per cabinet.

6 Model Update Requested

Now Quality Cabinets is considering renting better painting equipment and has asked us to update our model to help with this decision. The equipment will cost \$300 per week, but will reduce the time required to to do the painting by 15 minutes for a standard cabinet, by 30 minutes for a deluxe cabinet, and by 1 hour for an enhanced cabinet.

We decide to add a binary variable z. In the solution, if $z = 1$, then Quality Cabinets should rent the equipment. If $z = 0$, they should not.

6. Should the objective function change? If not, explain. If so, write the updated objective function.

$\max \; 25x_a + 45x_d + 60x_e \underbrace{- 300z}_{\text{fixed cost of buying machine}}$

7. If the equipment is not rented, what should the painting constraint be? Label it (A).

If equipment not rented → z=0 → Constraint doesn't change

(A) $2x_a + 4x_d + 5x_e \leq 700$ ← Same constraint as before

8. If the equipment is rented, what should the painting constraint be? Label it (B).

Equipment rented $z=1$

$x_a \rightarrow$ 15 min

$x_d \rightarrow$ 30 min

$x_e \rightarrow$ 1 hr

$1.75x_a + 3.5x_d + 4x_e \le 700$ (B)

Goal: If $z=0$ use constraint A

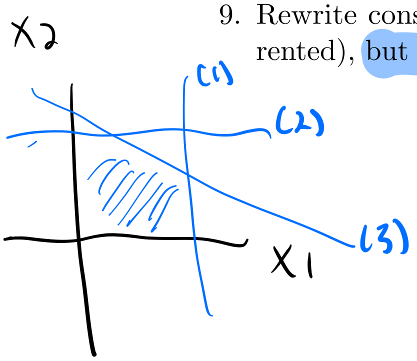
If $z=1$ use constraint B

7 Logical (Either/Or) Constraints

There is no place for "if – then" statements in a math programming model, so we have to enforce this logic indirectly using linear constraints and binary variables.

> A constraint is *relaxed* if it has no impact. We can relax a "$\le$" constraint by making the value on the right so large that it will never restrict the values of the variables on the left.

9. Rewrite constraint (A) above *using* z so that it is enforced if $z = 0$ (the equipment is not rented), but *relaxed* if $z = 1$ (the equipment is rented).



To relax, push the constraint so that is has no effect on feasible region

(A): $2x_a + 4x_d + 5x_e \le 700$

$z=1$ $\le 700 + M$ relaxed

$z=0$ ≤ 700 enforced

Relax (A): $2x_a + 4x_d + 5x_e \le 700 + Mz$

10. Rewrite constraint (B) above *using* z so that it is enforced if $z = 1$ (the equipment is rented), but *relaxed* if $z = 0$ (the equipment is not rented).

(B) $1.75x_a + 3.5x_d + 4x_e \le 700$

Relax B: $1.75x_a + 3.5x_d + 4x_e \le 700 + M(1-z)$

$\hookrightarrow$ $z=0$ $\le 700 + M$ Relaxed

$z=1$ ≤ 700 Enforced

11. Write the updated version of the partial Quality Cabinets model here.

$$\max \quad 25x_a + 45x_d + 60x_e - 300z$$

$$\text{s.t.} \quad 2x_a + 4x_d + 5x_e \le 700 + M \cdot z \qquad (A)$$

$$1.75x_a + 3.5x_d + 4x_e \le 700 + M(1-z) \qquad (B)$$

$z=0$ Don't rent Pay 0 (A) enforced (B) Relaxed

$z=1$ Do rent Pay 300 (A) Relaxed (B) enforced

8 Logical Constraints Summary

Suppose $a^T x \le b$ is a linear constraint, M is a number that is bigger than $a^T x$ could ever be (but not TOO big!–choose M wisely), and z is a binary variable.

A constraint that is enforces $a^T x \le b$ if $z = 0$ and relaxes $a^T x \le b$ if $z = 1$:

$$\boxed{a^T x \le b + M \cdot z}$$

A constraint that is enforces $a^T x \le b$ if $z = 1$ and relaxes $a^T x \le b$ if $z = 0$:

$$\boxed{a^T x \le b + M(1-z)}$$

9 Many more possibilities

There are many ways to creatively use linear constraints to enforce modeling requirements; this lesson contains only a few examples. Often this process takes some trial and error. **Always be sure to test the logic with various values of the decision variables to make sure the constraint is doing what you want it to do.**