Recall
\[ x_j = \begin{cases} 1 & \text{if player } j \text{ starts} \\ 0 & \text{o.w.} \end{cases} \]

Logical constraints (if-then and either-or constraints)

3. \( x_3 + x_6 \leq 1 \) (if player 3 starts, player 6 cannot)
or (at most one of players 3 & 6 can start)

If we write \( x_3 + x_6 = 1 \) instead, we are insisting that exactly one of them must start.

4. \( x_4 \geq x_1, \quad x_5 \geq x_1 \) (if player 1 starts, players 4 and 5 must start)

Again, we do not want to write \( x_4 = x_1 \) & \( x_5 = x_1 \) here. These constraints model "either players 1, 4, and 5 all start, or all do not start."

Alternatively, we could write
\[ x_4 + x_5 \geq 2x_1 \]
(by aggregating or adding both constraints)

Check: \( x_1 = 1 \Rightarrow x_4 + x_5 \geq 2 \)
\[ \Rightarrow x_4 = x_5 = 1 \]
\( x_1 = 0 \Rightarrow x_4 + x_5 \geq 0, \) which is redundant
5. \( x_2 + x_3 \geq 1 \) (either player 2 or player 3 or both must start) \( \checkmark \) this constraint allows both of them starting

\( x_2 + x_3 = 1 \) forces that exactly one of players 2 and 3 but not both must start.

by default, assume OR, and not exclusive-OR (XOR) unless specified otherwise. So use \( \geq \) by default, and not \( = \).

Objective function

\[
\text{max } Z = 3x_1 + 2x_2 + 2x_3 + x_4 + 3x_5 + 3x_6 + x_7 \quad \text{(total defense)}
\]

Prob 3. WV-1MP pg 502 503 fixed cost/charge problem

3 A manufacturer can sell product 1 at a profit of $2/unit and product 2 at a profit of $5/unit. Three units of raw material are needed to manufacture 1 unit of product 1, and

6 units of raw material are needed to manufacture 1 unit of product 2. A total of 120 units of raw material are available. If any of product 1 is produced, a setup cost of $10 is incurred, and if any of product 2 is produced, a setup cost of $20 is incurred. Formulate an IP to maximize profits.

decisions 1. how many of each product to make?
     2. set up costs — do we make any of products 1 and 2 at all?
\( x_j = \# \text{ units of product } j \text{ made, } j=1,2 \)
\( x_j \geq 0, \text{ not necessary to insist on being integers} \)
\( y_j = \begin{cases} 1 & \text{if } x_j > 0 \quad (\text{if any of product 1 is made}) \\ 0 & \text{o.w.} \end{cases} \)

**Constraints**

\[ 3x_1 + 6x_2 \leq 120 \quad (\text{raw math. limit}) \]

**Objective function**

\[ \max \ Z = 2x_1 + 5x_2 - 10y_1 - 20y_2 \quad (\text{profit}) \]

We need to model the relationship between \( x_j \) and \( y_j \):

\[ x_1 \leq M_1 y_1 \quad (\text{forcing constraints}) \]
\[ x_2 \leq M_2 y_2 \quad M_1, M_2 \text{ are big positive numbers.} \]

We want to use the smallest \( M_1 \) and \( M_2 \) that work.

If \( x_1 > 0 \), it can satisfy \( x_1 \leq M_1 y_1 \) only with \( y_1 = 1 \).

But if \( x_1 = 0 \), \( y_1 \) could be 0 or 1, just based on this constraint. But, the coefficient of \( y_1 \) in the max objective function is \(-10\), and hence \( y_1 \) is set to 0 in this case.

Similar explanation for \( x_2 \approx y_2 \) relationship.
Smallest values of $M_1$ and $M_2$

\[ M_1 = \frac{120}{3} = 40 \]
\[ M_2 = \frac{120}{6} = 20 \]

the max # of products 1 and 2 that could be made.

With $y_1 = 1$, the forcing constraint for product 1 reads

\[ x_1 \leq M_1, \text{ i.e., it specifies an upper bound on the } \]
\[ \# \text{ units product 1 that could be made.} \]

The complete MILP is presented here:

\[
\begin{align*}
\text{max } \quad Z &= 2x_1 + 5x_2 - 10y_1 - 20y_2 \\
\text{s.t. } \\
3x_1 + 6x_2 &\leq 120 \quad \text{(raw mate)} \\
x_1 &\leq 40y_1 \quad \text{(forcing constraint 1)} \\
x_2 &\leq 20y_2 \quad \text{(forcing constraint 2)} \\
x_1, x_2 &\geq 0, \quad y_1, y_2 \in \{0,1\} \\
\text{or } \\
y_1, y_2 &\text{ binary}
\end{align*}
\]

If we do not have a way to estimate good (i.e., small) values of $M_1$ and $M_2$ that work, we could use large numbers, e.g., $M_1 = M_2 = 10^6$, for instance. In practice, though, really large values of $M_1, M_2$ could make the problem harder to solve.
Facility location problem (from the project)

- $\bigcirc$ → candidate facility location (depot)
- $\times$ → customer
- $\bullet$ → facility opened in candidate location $j$ ($y_j = 1$)
- $\times \rightarrow \bullet$ customer $i$ assigned to facility $j$ ($x_{ij} = 1$)

Definitions:

$$y_j = \begin{cases} 1 & \text{if facility located in depot location } j \\ 0 & \text{otherwise} \end{cases}, \quad j = 1, \ldots, n$$

$$x_{ij} = \begin{cases} 1 & \text{if customer } i \text{ is assigned to facility in location } j \\ 0 & \text{o.w.} \end{cases}, \quad i = 1, \ldots, m, j = 1, \ldots, n$$

The aggregate and disaggregate models differ in how we enforce the relationship between the sets of binary variables $x_{ij}$ and $y_j$. The restriction we want to model is the following.

We can assign any customer to the facility in location $j$ only if we indeed locate (or open) a facility in the candidate location $j$.
The aggregate model:
\[ \sum_{i=1}^{m} x_{ij} \leq m y_j \quad \text{for } j = 1, \ldots, n \]

We could possibly assign up to all of the m customers to the facility opened in location \( j \) — hence \( m \) could be used in place of the big-M here.

The disaggregate model:
\[ x_{ij} \leq y_j \quad \text{for } i = 1, \ldots, m, \quad j = 1, \ldots, n \]

We force the relationships separately for each customer-location pair \((i, j)\) here. The big-M can hence be chosen as 1 here.

There are \( n \) constraints (one for each location) in the aggregate model, while there are \( mn \) constraints in the disaggregate model. But the disaggregate constraints are tighter, as all big-M values are 1, the smallest possible. The difference becomes large when \( m \) (and \( n \)) are large — and you can see it in the computation!