1) Class: Discussions on OptMap and SAT.
2) B&B & IP
3) Architecture Team for B&B.

**Divide & Conquer:**

Consider the following Optimization Problem:

\[
x^* = \arg \min_x \{ f(x) : x \in S \}
\]

**Proposition 1:** (Branching: \( b \) = Branching Factor)

Let \( S = S_1 \cup S_2 \cup \ldots \cup S_b \) be a decomposition of \( S \) into smaller sets (\( \forall k \mid |S_k| < |S| \) \( S_k \neq \emptyset \))\( \forall k \neq l \).

Let \( x^*_k = \arg \min_x \{ f(x) : x \in S_k \} \), \( \forall k \in \{1, \ldots, b\} \).

Then

\[
f(x^*) = \min \{ f(x^*_1), \ldots, f(x^*_b) \}.
\]

**Example:**

\( S = \{0,1\}^n \)

\[ S_0 = \{ x \in S \mid x_i = 0 \} \quad \text{and} \quad S_1 = \{ x \in S \mid x_i = 1 \} \]

\[ S_{00} = \{ x \in S \mid x_1 = 0, x_2 = 0 \} = \{ x \notin S_0 \mid x_2 = 0 \} \]

\[ S_{01} = \{ x \in S \mid x_1 = 0, x_2 = 1 \} = \{ x \in S_0 \mid x_2 = 1 \} \]

\[ S_{10} = \{ x \in S \mid x_1 = 1, x_2 = 0 \} = \{ x \in S_1 \mid x_2 = 0 \} \]
Binary Enumeration Tree.

**Proposition 2:** (Bounding & Pruning ([Fathoming]): Upper and Lower Bounds.)

Let \( S = S_1 \cup S_2 \cup \ldots \cup S_b \) be a decomposition of \( S \) into smaller sets:

\[ \forall k \; S_k \neq \emptyset \; \text{and} \; \cup S_k \subseteq S \]

Let \( x^*_k = \arg\min_x \{ f(x) \mid x \in S_k \} \; \forall k \in \{1, \ldots, b\} \)

Let \( f_k^{\text{low}} \leq f(x^*_k) \leq f_k^{\text{up}} \) be lower and upper bounds on \( x^*_k \).

Let \( f^{\text{low}} = \min_k f_k^{\text{low}} \) and \( f^{\text{up}} = \min_k f_k^{\text{up}} \).

Then:

1a) \( f^{\text{low}} \leq f(x^*) \leq f^{\text{up}} \)

1b) \( f^{\text{low}} = f^{\text{up}} \Rightarrow f^{\text{low}} = f^{\text{up}} = f(x^*) \)

2) If \( S_t = \{ x_t^* \} \) be a singleton set, then

\[ f(x_t) = f(x_t^*) = f_t^{\text{low}} = f_t^{\text{up}} \]

3) **PRUNE/FATHOM**

\[ f(x^*) = \min \{ f(x_t^*) \mid x_t^* \in S_t, \; f_t^{\text{low}} \leq f_t^{\text{up}} \} \]
\[\text{Divide & Conquer:}\]

Let \( S = S_0 \cup S_1 \cup \ldots \cup S_b \) and temp := MIN (temp, OptDC \( f, S_0 \)).

For \( i = 1, \ldots, b \) loop

\[\text{if } S_i \neq \emptyset \text{ then}\]

Select \( x \in S_i \), arbitrarily;

temp := \( \langle x, f(x) \rangle \);

Let \( S' = S \setminus \{x\} \);

[\( s, f, s' \) \( \notin S \) \\
return temp;]

else

OptDC \( f, S \) := singleton

then return \( \langle x, f(x) \rangle \);

end if

end loop

\[\text{subject not explored!}\]
Branch & Bound:

\[ \text{OptBB}(f, S); \]

\text{if } S = \{ x \} = \text{singleton then return } \langle x, f(x) \rangle \]

\text{else}

\[ S = S_1 \cup S_2 \cup \ldots \cup S_b; \quad [S_i \subseteq S \land S_i \neq \emptyset] \]

\[ f^{\text{up}} = \min (f_1^{\text{up}}, f_2^{\text{up}}, \ldots, f_b^{\text{up}}); \]

\[ f^{\text{low}} = \min (f_1^{\text{low}}, f_2^{\text{low}}, \ldots, f_b^{\text{low}}); \]

Select \( x \in S \), arbitrarily;

\[ \text{temp} := \langle x, f(x) \rangle \]

for \( i = 1, \ldots, b \) loop

\text{if } S_i \neq \emptyset \text{ and } f_i^{\text{low}} \leq f^{\text{up}} \text{ then}

\[ \text{temp} := \text{MIN} (\text{temp}, \text{OptBB}(f, S_i)); \]

return temp.

---

How to compute Upper and Lower Bounds?

\text{a) If } S = \{ x \} = \text{singleton} \quad \text{or } |S| = k, \text{ small} \]

\[ f^{\text{up}} = f(x) = f^{\text{low}} \]

\[ f^{\text{up}} = \min (f(x_1), \ldots, f(x_k)) \]

\text{b) If } |S| > k \text{ then}

\[ f^{\text{up}} \leq f(x), \forall x \in S. \]

Randomly sample \( S, k \) times; \( f(x_1), \ldots, x_k \)

Set \[ f^{\text{up}} = \min (f(x_1), \ldots, f(x_k)) \]

\text{c) How to compute lower Bounds.}
TSP (Travelling Salesman Problem)

\[ f_{1/2}^\text{low} \left( G, w \right) = \frac{1}{2} \sum_{i \in V} w_{ij} + w_{ik} \]

\[ w_{ij} = \min \left( W_{i,j} \right) \]

\[ w_{ik} = \min \left( W_{i,k} \setminus w_{ij} \right) \]

Sum of the costs of the two least cost edges adjacent to \( i \).

Note: Cost of any tour

\[ = \frac{1}{2} \sum_{i \in V} W_{i,i+1} + W_{i,i+2} \]

Sum of the costs of the two tour edges adjacent to \( i \).

0.1 Integer Programming:

\[
\begin{align*}
\max & \quad \sum_{j=1}^{m} v_j x_j \\
\sum_{j=1}^{m} w_j x_j & \leq b \\
x_j & \in \{0, 1\} \quad j = 1, \ldots, n \\
\end{align*}
\]

Relaxing \( x_j \)'s will result in

\[ 0 \leq x_j \leq 1 \quad j = 1, \ldots, n \]

A polytime linear programming problem.
The Knapsack Problem: (Binary KP).

1) A set of objects to select from.
   \( n \): Number of objects, indexed by
   \( 1, 2, \ldots, j, \ldots \).

2) Each object has a weight \( w_j \), \( j \in \{1, \ldots, n\} \)
   and a value \( v_j \).

3) You have a knapsack to carry the objects, but
   you are not able to carry more than a weight
   of \( b \).

   \[ x_j = \text{Indicator variable} = \begin{cases} 1 & \text{if object } j \text{ is} \\
0 & \text{otherwise.} \end{cases} \]

4) Your goal is to maximize the total value you can carry.

   \[ \max \sum_{j=1}^{n} w_j x_j \quad \text{subject to} \quad \sum_{j=1}^{n} v_j x_j \leq b \]

   \[ x_j \in \{0, 1\}, j = 1, \ldots, n. \]

- \( \)

- Integer Linear Programming:
  \[ \max c^T x \quad c \in \mathbb{Z}^n \]
  subject to \[ Ax \leq b \quad A \in \mathbb{Z}^{m \times n} \]
  \[ x \geq 0 \quad b \in \mathbb{Z}^m \]
  \[ \text{and } x \in \mathbb{Z} \]
0–1 Case. LP relaxation

Solve the LP version of the problem:

\[ x \in \{0, 1\}^n \text{ is replaced by } 0 \leq x \leq 1. \]

Let \( x^* \) be a solution to the relaxed problem \( x^* \in \mathbb{R}^n \subseteq [0, 1]^n \), though the polytope desired must be in \( \times 0, 1^2 \).

Upper bound.

\[
\begin{align*}
& x_i \text{ in } \mathbb{R}^n \\
& \{ \\
& \quad (a) \text{ Round each } x_i^* \text{ to } 0 \text{ or } 1. \\
& \quad (b) \text{ Generate random } x_i \in \{0, 1\} \text{ with } \text{Prob}(x_i = 1) = x_i^* \\
& \quad \text{Check that the guess generated } \hat{x}_i \text{ is feasible.} \\
& \end{align*}
\]

Suppose you have \( \hat{x}_1, \hat{x}_2, \ldots, \hat{x}_k \) as feasible guesses.

\[
A\hat{x}_i \leq b. \quad \text{feasibility}
\]

Upper bound = \( \min (+\infty, c^T \hat{x}_i) \)

Lower bound

\[ c^T x^* \text{ is a lower bound on the problem} \]

since every solution to ILP is also a solution to the relaxed LP.