Branch and bound

OOptimization problem: Seeks to minimize or maximize some objective function

o feasible solution: a point in the problems search space that satisfies all the problems constraines

O optimal solution: a feasible solution with the best value of the objective function.

- a way to provide, for every node of a state-space tree, a bound on the best value of the objective function[1] on any solution that can be obtained by adding further components to the partially constructed solution represented by the node
- the value of the best solution seen so far

-> If the bound value is not the best solution so far, the node is nonpromising and can be terminated.

In general, we terminate a search path at the current node in a state-space tree of a branch-and-bound algorithm for any one of the following three reasons:

- The value of the node's bound is not better than the value of the best solution seen so far.
- The node represents no feasible solutions because the constraints of the problem are already violated.
- The subset of feasible solutions represented by the node consists of a single point (and hence no further choices can be made)—in this case, we compare the value of the objective function for this feasible solution with that of the best solution seen so far and update the latter with the former if the new solution is better.

o Assignment problem

-> n people to do n jobs for as cheap as possible.

$$C = \begin{array}{c} \\ \\ \\ \\ \end{array}\begin{array}{cccc} \text{job 1} & \text{job 2} & \text{job 3} & \text{job 4} \end{array}$$

$$C = \begin{bmatrix} 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 8 \\ 7 & 6 & 9 & 4 \end{bmatrix} \begin{array}{l} \text{person } a \\ \text{person } b \\ \text{person } c \\ \text{person } d \end{array}$$

-> smallest is $2+3+1+4=10$

↳ not legit, 2 & 3 same column

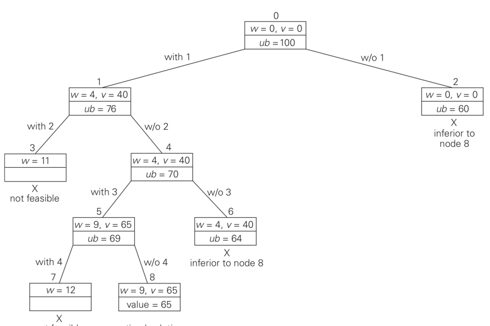
-> smallest legit: $9+3+1+4=17$

-> How to construct state-space tree?

live: still promising leaves.

Best first branch and bound: a later bound can still be better than a promising leaf.

o knapsack problem:



item	weight	value	value/weight
1	4	$40	10
2	7	$42	6
3	5	$25	5
4	3	$12	4

The knapsack's capacity W is 10.