Branch-and-Bound

- Typically used to look for an optimal solution by searching an implicit graph
  - The graph is implicit and typically acyclic or even a tree
- During the search, we calculate at each node a bound on the possible value of any solution that might lie further in the graph
  - If the bound guarantees us the further solution is worse than the best solution so far, we need not to explore this part of graph anymore
- Implementation
  - Priority list

The assignment problem

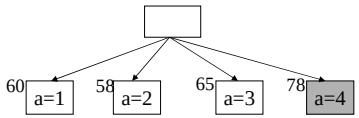
- n agents are to be assigned n tasks. Each agent has exactly one task to perform
  - If agent i, $1 \le i \le n$, is assigned task j, $1 \le j \le n$, the cost to perform the task is c_{ij}.
  - Given the complete matrix of costs, assign the agents to tasks so as to minimize the total cost of executing the n tasks
- Can you come out a brutal-force algorithm? How much does it cost?
  - Some similarity to the 8 queens problem, but we need consider the cost

Some facts about the problem

- Some initial bounds
  - One upper bound
    - total cost along diagonals: min(73, 87) = 73
  - One lower bound
    - Sum of minimum cost on each row (best job): 58
    - Sum of minimum cost on each row (best person): 49
  - We have an initial bound [58, 73]

	1	2	3	4
a	11	12	18	40
b	14	15	13	22
c	11	17	19	23
d	17	14	20	28

Example

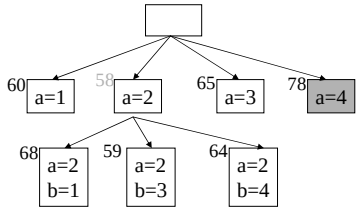


Initial bound [58..73]

- At a time, we expand a node and estimate the minimum possible cost, using the heuristic that each job has to be assigned to an agent.
- We disable a node if its cost is higher than the current upper bound

	1	2	3	4
a	11	12	18	40
b	14	15	13	22
c	11	17	19	23
d	17	14	20	28

Example

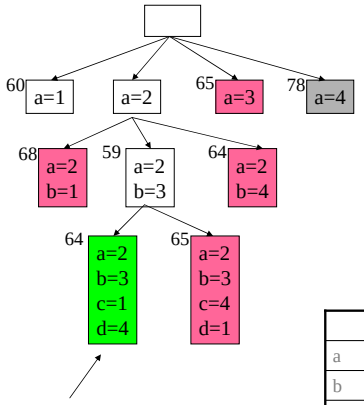


Initial bound
[58..73]

- Expands the most promising node

	1	2	3	4
a	11	12	18	40
b	14	15	13	22
c	11	17	19	23
d	17	14	20	28

Example



Initial bound
[58..73]

Current bound [58..64].

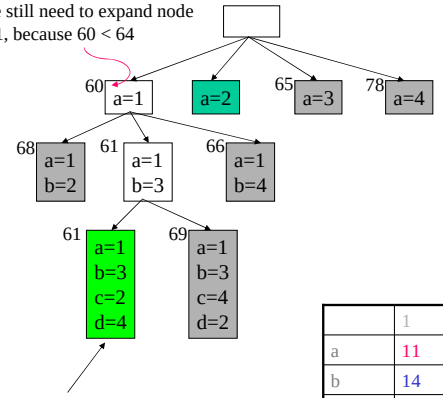
- The pink nodes are disabled due to this bound

The best solution
so far. New upper bound

	1	2	3	4
a	11	12	18	40
b	14	15	13	22
c	11	17	19	23
d	17	14	20	28

Example

We still need to expand node
a=1, because 60 < 64



The best solution

	1	2	3	4
a	11	12	18	40
b	14	15	13	22
c	11	17	19	23
d	17	14	20	28