Informed State Space Search

COURSE: CS60045

Pallab Dasgupta
Professor,
Dept. of Computer Sc & Engg

BASICS OF HEURISTIC SEARCH

STATE or CONFIGURATION:
- A set of variables which define a state or configuration
- Domains for every variable and constraints among variables to define a valid configuration

STATE TRANSFORMATION RULES or MOVES:
- A set of RULES which define which are the valid set of NEXT STATE of a given State
- It also indicates who can make these Moves (OR Nodes, AND nodes, etc)

STATE SPACE or IMPLICIT GRAPH
- The Complete Graph produced out of the State Transformation Rules.
- Typically too large to store. Could be Infinite.

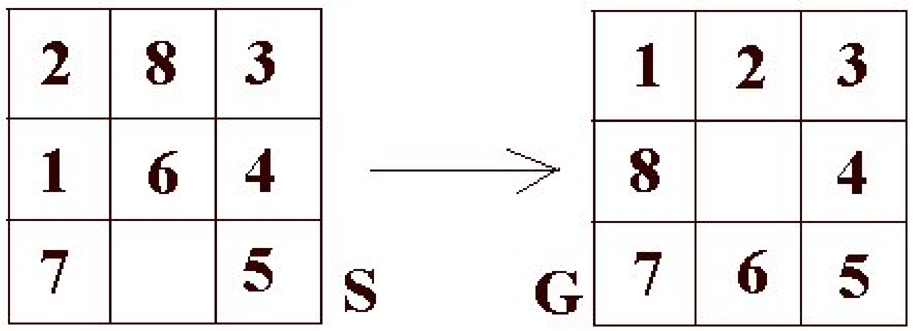
INITIAL or START STATE(s), GOAL STATE(s)

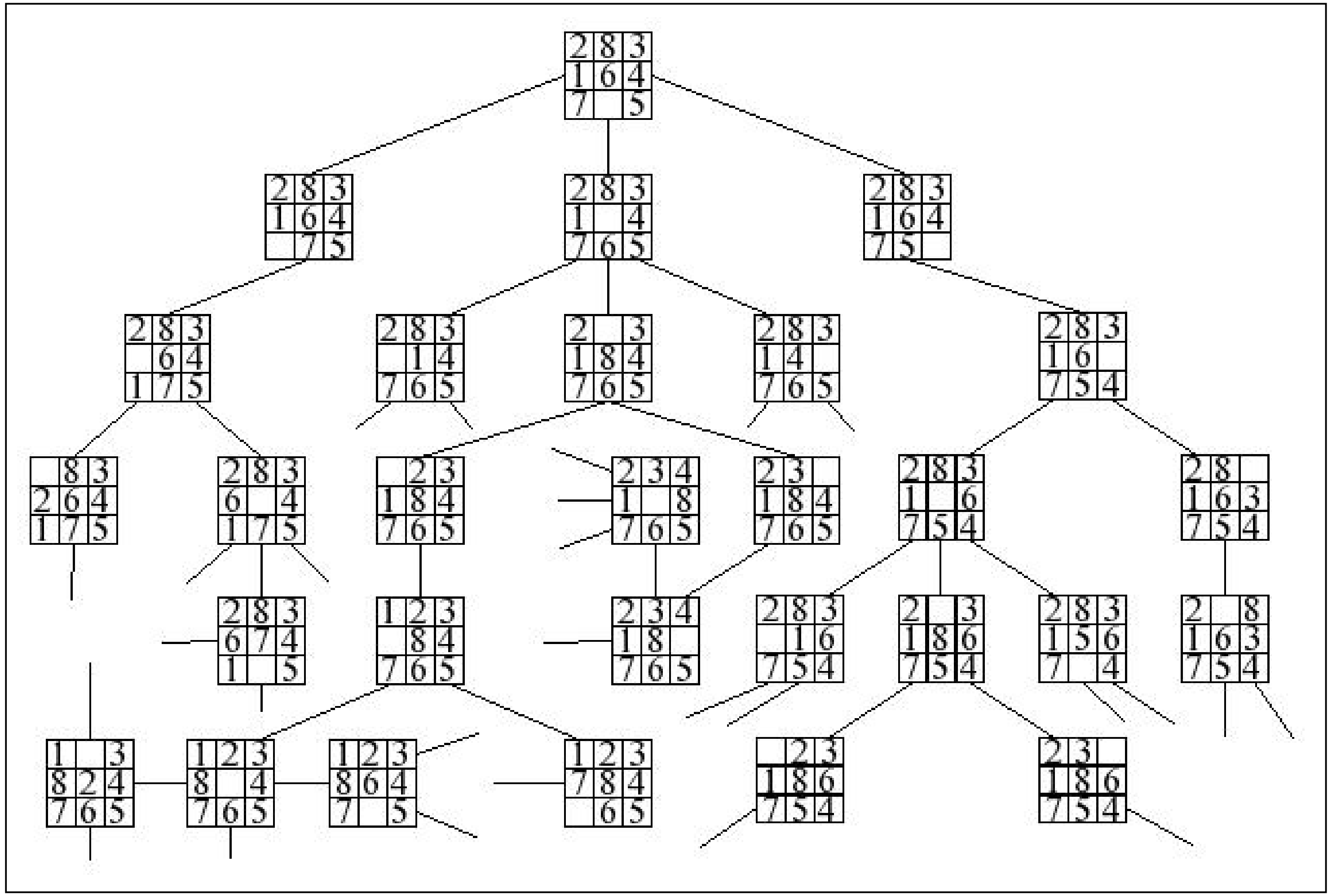
SOLUTION(s), COSTS
- Depending on the problem formulation, it can be a PATH from Start to Goal or a Sub-graph of And-ed Nodes

HEURISTICS
- Estimates of cost from a given state to goal. This, along with the current cost of the path from start till now is used to guide the search

HEURISTIC SEARCH ALGORITHMS
- Algorithm A*, Depth-First Branch & Bound, IDA*, AO*, Alpha-Beta, etc
- Knowledge vs Search

Example of Heuristics: 8 puzzle problem

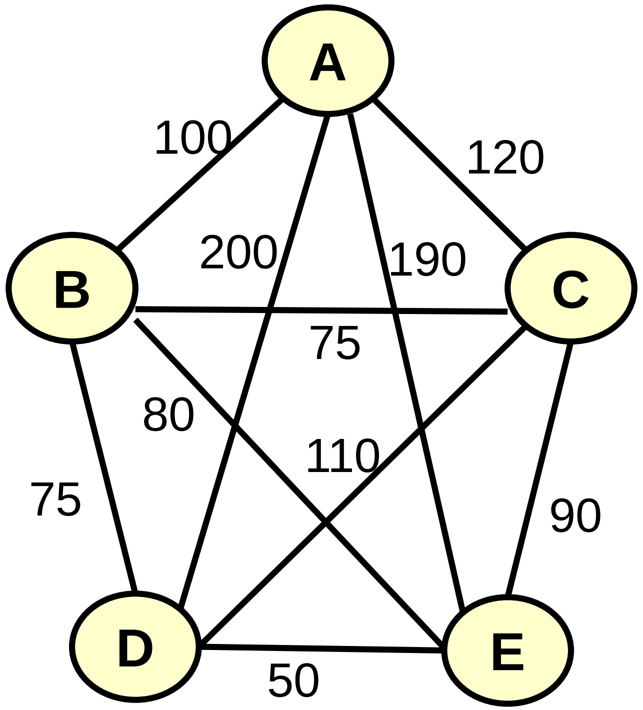


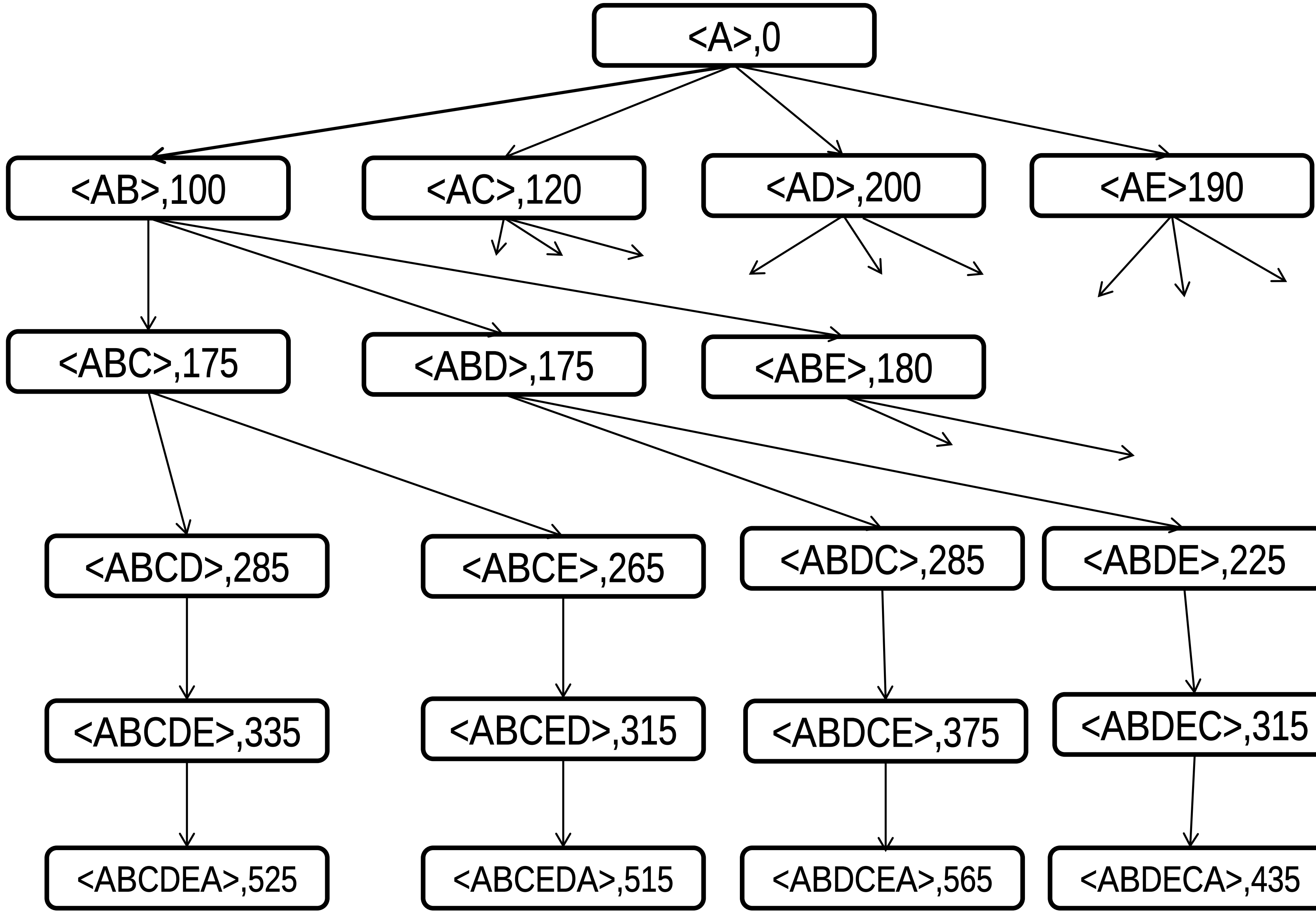


Heuristic 1: Number of misplaced tiles

Heuristic 2: Sum of tile distances from goal

Travelling Salesperson Problem: Heuristics

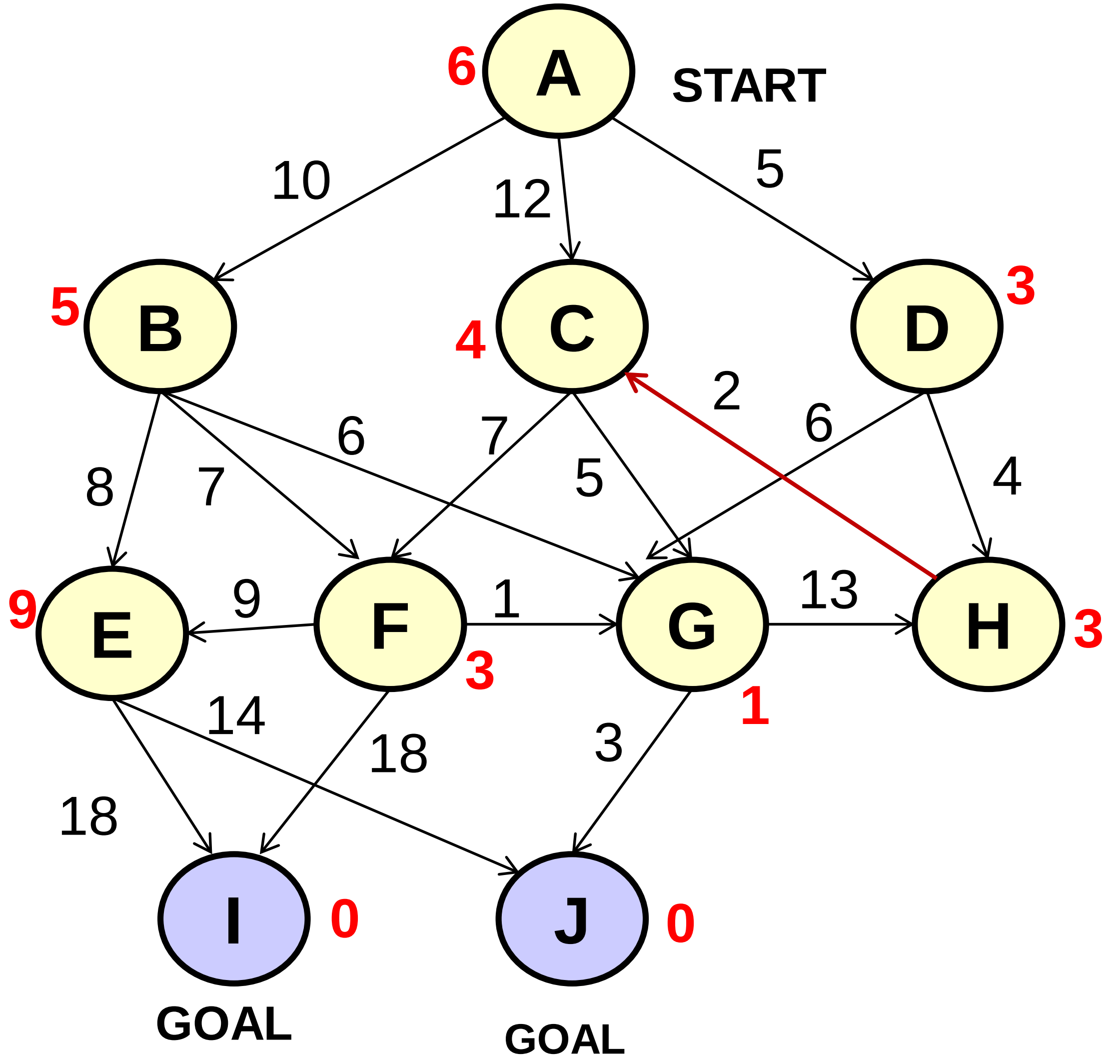




Heuristic 1: Cost of shortest path from current node to start through remaining nodes only

Heuristic 2: Cost of minimum cost spanning tree of remaining nodes

Searching State Spaces with Edge costs, Heuristic estimates



- HEURISTIC SEARCH ALGORITHMS:
  - DFBB
  - A*: Best First Search,
  - IDA*: Iterative Deepening A*
  - Every edge (n, m) in the graph has a cost c(n,m) > 0.
  - **HEURISTIC Estimates: h(n) ≥ 0 at** every node is the estimated cost of the minimum cost path from node n to goal

- PROPERTIES
  - SOLUTION GUARANTEES
  - MEMORY REQUIREMENTS

Algorithm A*

1. Initialize: Set OPEN = {s},
   CLOSED = { }, $g(s) = 0$, $f(s) = h(s)$

2. Fail: If OPEN = { }, Terminate & Fail

3. Select: Select the minimum cost state, n, from OPEN. Save n in CLOSED

4. Terminate: If $n \in G$, terminate with success, and return $f(n)$

5. Expand:
   For each successor, m, of n
   - If $m \notin [OPEN \cup CLOSED]$
     - Set $g(m) = g(n) + C(n,m)$
     - Set $f(m) = g(m) + h(m)$
     - Insert m in OPEN
   - If $m \in [OPEN \cup CLOSED]$
     - Set $g(m) = \min \{ g(m), g(n) + C(n,m) \}$
     - Set $f(m) = g(m) + h(m)$
     - If $f(m)$ has decreased and $m \in CLOSED$, move m to OPEN

6. Loop: Go To Step 2.

Algorithm A*

A heuristic is called admissible if it always under-estimates, that is, we always have $h(n) \leq f^*(n)$, where $f^*(n)$ denotes the minimum distance to a goal state from state n.

- For finite state spaces, A* always terminates
- At any time time before A* terminates, there exists in OPEN a state n that is on an optimal path from s to a goal state, with $f(n) \leq f^*(s)$
- If there is a path from s to a goal state, A* terminates (even when the state space is infinite)
- Algorithm A* is admissible, that is, if there is a path from s to a goal state, A* terminates by finding an optimal path
- If A_1 and A_2 are two versions of A* such that A_2 is more informed than A_1, then A_1 expands at least as many states as does A_2.
    - If we are given two or more admissible heuristics, we can take their max to get a stronger admissible heuristic.

Execution of A*

OPEN = {A[0,6,6]}, CLOSED = {}

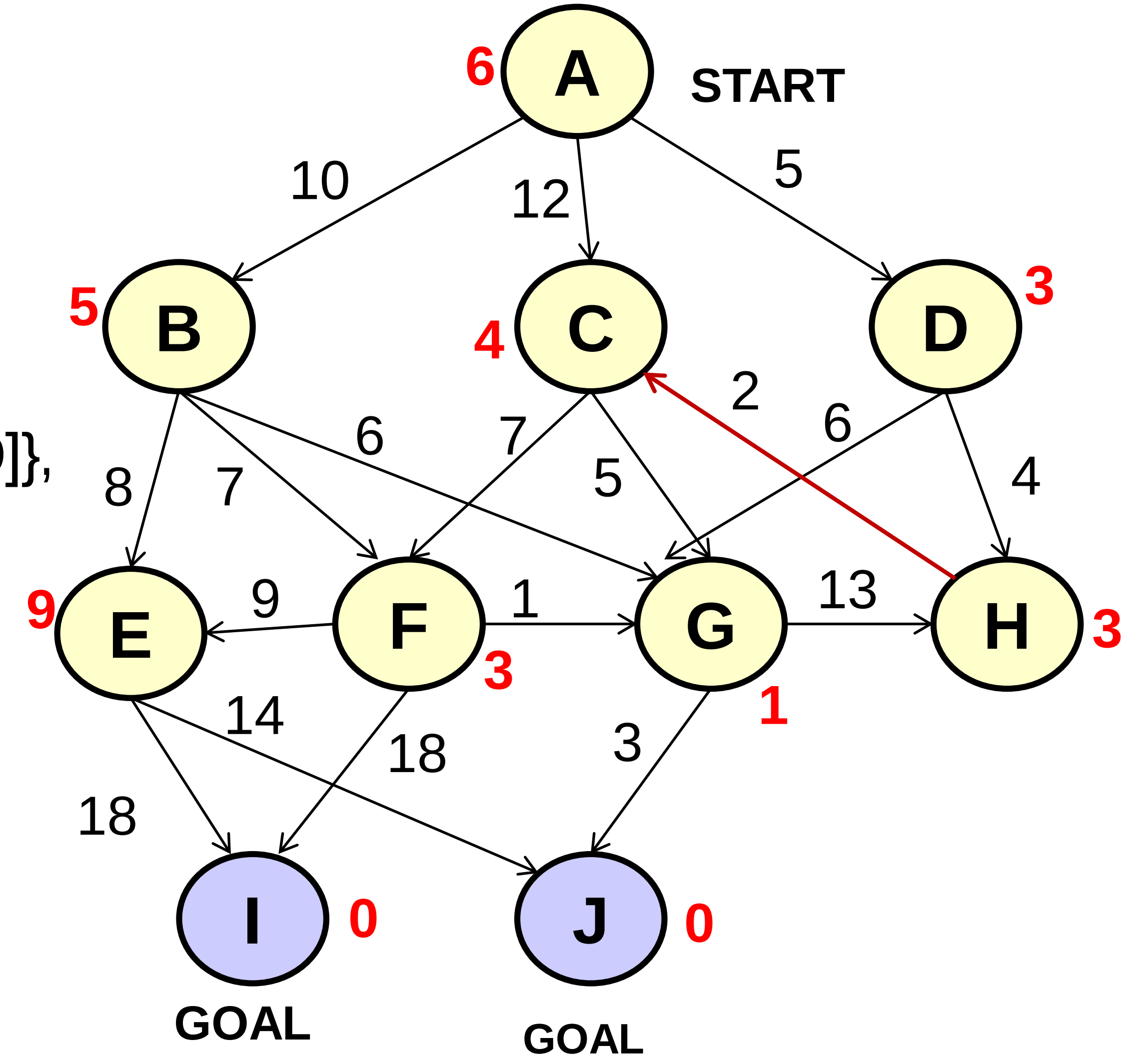
OPEN = {B[10,5,15,A], C[12,4,16,A], D[5,3,8,A]},
CLOSED = {A}

OPEN = {B[10,5,15,A], C[12,4,16,A], G[11,1,12,D], H[9,3,12,D]},
CLOSED = {A,D}

OPEN = {B[10,5,15,A], C[11,4,15,H], G[11,1,12,D]},
CLOSED = {A,D,H}

OPEN = {B[10,5,15,A], C[11,4,15,H], J[14,0,14,G]},
CLOSED = {A[],D[A],H[D],G[D]}

Goal J found. Terminate with cost 14 and path A,D,G,J.



Other Algorithms: DFBB, A*

DEPTH FIRST BRANCH AND BOUND (DFBB)

1. Initialize Best-Cost to f(s)
2. **Perform DFS with costs and Backtrack from any node n whose f(n) ≥ Best**-Cost
3. On reaching a Goal Node, update Best-Cost to the current best
4. Continue till OPEN becomes empty

ITERATIVE DEEPENING A* (IDA*)

1. Set Cut-off Bound to f(s)
2. Perform DFBB with Cut-off Bound. Backtrack from any node whose f(n) > Cut-off Bound.
3. If Solution is Found, at the end of one Iteration, Terminate with Solution
4. If Solution is not found in any iteration, then update Cut-off Bound to the lowest f(n) among all nodes from which the algorithm Backtracked.
5. Go to Step 2

PROPERTIES OF DFBB AND IDA*: Solution Cost, Memory, Node expansions, Heuristic Accuracy, Performance on Trees / Graphs

Iterative Deepening A*: *bounds*

- In the worst case, only one new state is expanded in each iteration
  - If A* expands N states, then IDA* can expand:

$$1 + 2 + 3 + \ldots + N = O(N^2)$$

- IDA* is asymptotically optimal

Summary

- Heuristic Search Methods use Problem Specific Information in the form of HEURISTIC Estimate Functions
- The Fundamental Algorithms include Algorithm A* (Best-First Search), DFBB (Depth-First Branch and Bound) and IDA* (Iterative Deepening A*)
- These algorithms guarantee optimal cost solutions and work better with more accurate heuristic estimates
- They have varying memory requirements and node expansion counts depending on the structure of the State Space
- In the case of And/Or Graphs and Game Trees, Marking algorithms are used like Algorithm AO* and Alpha Beta Pruning.
- Other Form of Heuristics include Islands, Multiple Heuristics, Bi-Directional Search, etc
- Advanced memory and time-bounded algorithms are needed to manage search in limited space and time
- Multi-Objective Heuristic Search is needed for some problems where multiple cost functions are involved

Additional Topics: Monotone Heuristics

- An admissible heuristic function, h( ), is monotonic if for every successor m of n:

$$h(n) - h(m) \leq c(n,m)$$

- If the monotone restriction is satisfied, then A* has already found an optimal path to the state it selects for expansion.

- If the monotone restriction is satisfied, the f-values of the states expanded by A* is non-decreasing.

Additional Topics: Pathmax

- Converts a non-monotonic heuristic to a monotonic one:

  - During generation of the successor, m of n we set:

    $$h'(m) = \max \{ h(m), h(n) - c(n,m) \}$$

    and use h'(m) as the heuristic at m.

Additional Topics: Inadmissible heuristics

- Advantages:
  - In many cases, inadmissible heuristics can cause better pruning and significantly reduce the search time

- Drawbacks:
  - A* may terminate with a sub-optimal solution

Additional Topics: Memory bounded A*: MA*

- Whenever |OPEN ∪ CLOSED| approaches M, some of the least promising states are removed
- To guarantee that the algorithm terminates, we need to back up the cost of the most promising leaf of the subtree being deleted at the root of that subtree
- Many variants of this algorithm have been studied. Recursive Best-First Search (RBFS) is a linear space version of this algorithm

Multi-Objective A*: MOA*

- Adaptation of A* for solving multi-criteria optimization problems
  - Traditional approaches combine the objectives into a single one
  - In multi-objective state space search, the dimensions are retained

- Main concepts:
  - Vector valued state space
  - Vector valued cost and heuristic functions
  - Non-dominated solutions