3.3. A* Search: Minimization of the total estimated solution cost

a) Distinctive features

1) Widely known form of best-first search

2) Evaluation function,

$$f(n) = g(n) + h(n), \text{ where}$$

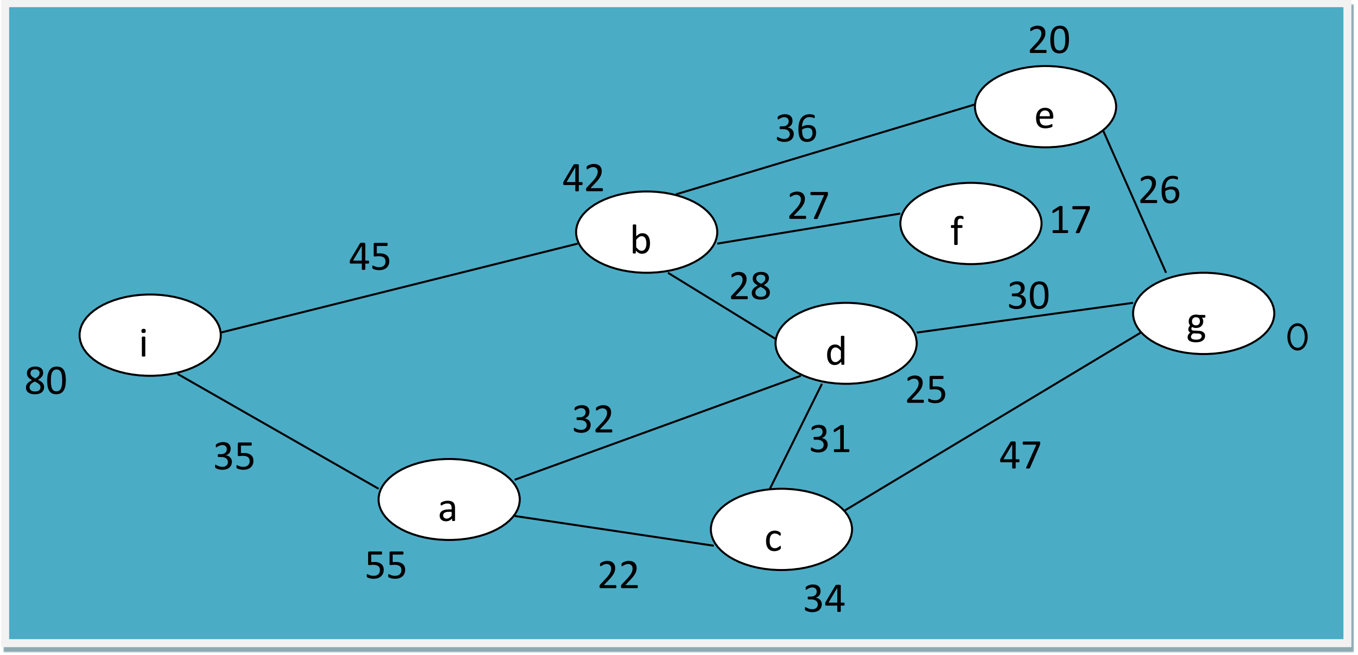
g(n) – an actual path cost from initial node to node n,
h(n) – estimated cost of the cheapest path from n to the goal,
f(n) – <u>estimated cost</u> of the <u>cheapest solution through node n</u>.

3) Generates all neighbors (may be repeatedly), and puts in PQ

4) Suboptimal solutions are avoided.

b) Execution of the algorithm

❖ The same problem instance is taken:



i – Initial state/node g – Goal state/node

❖ **Execution shown in the tree of visited nodes and their children:**

……….

c) Important decisions about A* search:

1. A* in tree search is optimal if h(n) is admissible.
2. A* in general graph search is optimal if h(n) is consistent.
3. If h(n) is consistent, then the values of f(n) along any path are non decreasing.
4. A* expands all nodes with f(n) < C*, where C* is the cost of the optimal solution.
5. A* is complete, assuming that there are only finite number of nodes with cost less than or equal to C*.
6. A* is said to be optimally efficient, that is, no other optimal algorithm is guaranteed to expand fewer nodes than A* does.

d) Problems:

1. Number of expanded nodes for most problems grows exponentially.

2. Admissible and consistent heuristics are very hard to find.

Remedies:

a) Various memory bounded search techniques (drop worst nodes when memory is full, …) are used.

b. Variants of A* do not insist on optimal solution, but find good solution (suboptimal) even with heuristics not strictly admissible.