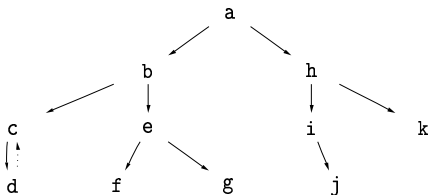


**Search**

**Depth-First Search:** finds solution path  $S_{ol}$  from given node  $N$  to some goal node (there can be several):

- if  $N$  is a goal node,  $S_{ol}=[N]$  else
- if there is a successor node  $N_1$  of  $N$  such that there is a path  $S_{ol1}$  from  $N_1$  to a goal node  $S_{ol}=[N, S_{ol1}]$



**Breadth-First Search:** finds solution path  $S_{ol}$  from given node  $N$  to some goal node:

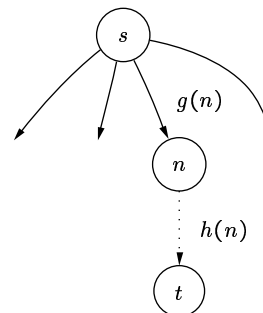
- if  $N$  is a goal node,  $S_{ol}=[N]$  else
- generate one-step extension of paths in candidate lists, adding extensions to that list.
- more detailed: given list of candidate paths
  - if first path contains goal node as head, solution
  - else
    - \* remove first path from candidate list
    - \* generate set of one-step extensions
    - \* append them to list of candidates
    - \* call breadth-first search on this list

**Heuristic Search**

**Heuristics:**

- consider a heuristic estimator  $f$
- $f(n)$  estimates “cost” of  $n$ , i.e. the cost of best solution path from start node  $s$  to some goal node, say  $t$ , provided that path goes via  $n$

$$f(n) = \underbrace{g(n)}_{\substack{\text{actual cost from } s \text{ to } n \\ \text{not necessarily optimal,} \\ \text{estimate of minimal cost} \\ \text{from } s \text{ to } n}} + \underbrace{h(n)}_{\substack{\text{guesswork!} \\ \text{no universal solution}}}$$



**Best-First Search:**

- breadth-first selects the shortest path
- best-first selects the least “costly” path

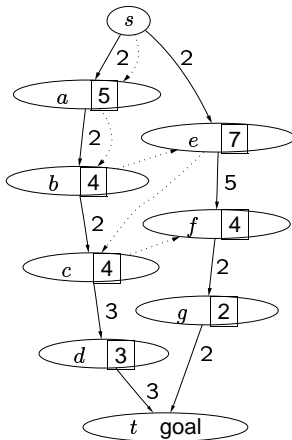
**For that:** define  $c(n, n')$  as cost for moving from node  $n$  to  $n'$

**Competing Subtrees:** among competitors, only one subtree is active at a time: the most promising, i.e. that with the lowest  $f$ -value

switch to alternative, if that changes

**Example:**  $f(X) = g(X) + h(X)$

Budget of subtree spent until exhausted, switching to other tree



5

## Task: Find Shortest Route

```

l(N, F/G)           % leaf
t(N, F/G, Subs)
  ^
  |
  ordered according to increasing f-values
    
```

Here we have:

- $G = g(N)$
- $F$  is updated  $f$ -value, value of most promising successor
- $f(T) = \min_i f(S_i)$

6

## Best-First Search

**Best-First Search:** finds cheapest solution path  $S_{ol}$  from given node  $N$  to some goal node:

1. if  $N$  is a goal node,  $S_{ol} = [N]$  else
2. initialize heap of candidate paths (sorted according to cost, *best first*), containing  $[N]$
3. pop head of heap (best candidate solution so far) as candidate solution  $C$
4. if  $C$  has node as head, solution
5. generate all one-step extensions of  $C$ , add them to heap
6. go to 3

7

## Admissibility

**Def.:** A search algorithm is *admissible* if it always produces an optimal solution.

**Note:** Above algorithm immediately produces an optimal solution if for each node  $n$ ,  $h^*(n)$  is the cost of an optimal path from  $n$  to some goal node.

**Note:** The Best-First algorithm is a variant of the noted  $A^*$  algorithm.

**Theorem:** Best-First is admissible if it uses an *optimistic* heuristic  $h$ , i.e.  $h$  with

$$h(n) \leq h^*(n)$$

**Default:** setting  $h(n) := 0$  is always possible, giving breadth-first search, but has no predictive power.

8