Review

Monday, September 19, 2016 9:03 AM

$f(n) = g(n) + h(n)$

rank states on the open list

A* – admissible heuristic

Handling shortcuts

```
WHILE (NOT isgoal(current) AND open ≠ NIL) DO:
    closed <— closed + {current}
    FOREACH n ∈ successors(current, ops) DO:
        IF n is not on open or closed THEN DO:
            compute g(n)
            Insert n into open (ranked on g())
        ELSE IF n is on open AND n is reached by a shorter path THEN DO:
            n.parent <— current
            update g(n)
            resort open
    END FOREACH
    current <— pop(open)
END WHILE
```

shortcuts

Best-First: h() can be admissible or not

Handling shortcuts for Best-First

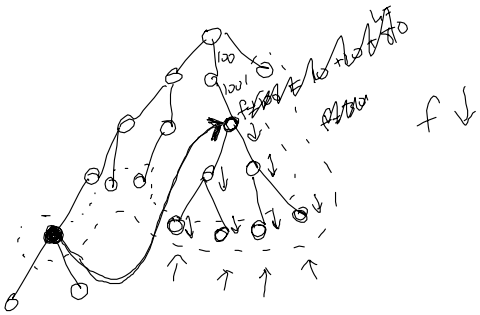
```
WHILE (NOT isgoal(current) AND open ≠ NIL) DO:
    closed <— closed + {current}
    FOREACH n ∈ successors(current, ops) DO:
        IF n is not on open or closed THEN DO:
            compute f(n)
            Insert n into open (ranked on g())
        ELSE IF n is on open AND not on closed AND n is reached by a shorter path THEN DO:
            n.parent <— current
            update f(n)
            resort open
        ELSE IF n is on closed AND n is reached by a shorter path THEN DO:
            n.parent <— current
            update f(n)
            Propagate new f-values to all descendants using DFS
                (stop generating successors if a node is on open)
    END FOREACH
    current <— pop(open)
```

shortcuts

hasn't generated successors yet

start w/ changed parent

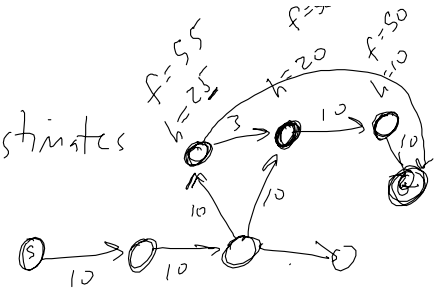
Depth bounded



END WHILE

Admissible

h() never overestimates



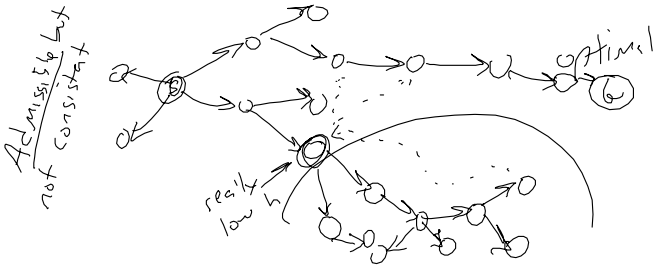
If you don't have admissibility → Non-optimal alg.

Consistency → Admissible

Admissible h under-estimates

Consistent h under-estimates too much growing up

$h(A) - h(B) \leq K(A,B)$



h(n) = 0 for all n ← Admissible

↳ ~~Breadth-first~~ Uniform-cost