AI SMPS 2023 Week 5 Algorithms

Prepared by S. Baskaran

Algorithms

A*(S)

```
1  default value of g for every node is +∞
2  parent(S) ← null
3  g(S) ← 0
4  f(S) ← g(S) + h(S)
5  OPEN ← S : []
6  CLOSED ← empty list

7  while OPEN is not empty
8      N ← remove node with lowest f value from OPEN
9      add N to CLOSED

10     if GOAL-TEST(N) = TRUE
11         return RECONSTRUCT-PATH(N)

12     for each M in MOVE-GEN(N)
13         if g(M) > g(N) + k(N, M)
14             parent(M) ← N
15             g(M) ← g(N) + k(N, M)
16             f(M) ← g(M) + h(M)

17             if M is in OPEN
18                 continue
19             else if M is in CLOSED
20                 PROPAGATE-IMPROVEMENT(M)
21             else add M to OPEN        ▷ M is new

22 return empty list
```

PROPAGATE-IMPROVEMENT(M)

```
23 for each X in MOVE-GEN(M)
24     if g(X) > g(M) + k(M, X)
25         parent(X) ← M
26         g(X) ← g(M) + k(M, X)
27         f(X) ← g(X) + h(X)

28         if X is in CLOSED
29             PROPAGATE-IMPROVEMENT(X)
30 return
```

For state space search, BnB maintains the pair (path,g) in the OPEN list, and expands the cheapest path in each iteration.

BnB(S)

```
1  OPEN ← (S : [], 0) : []        ▷ use priority queue
2  while OPEN is not empty
3      (path, g) ← head OPEN      ▷ (path,cost)
4      N ← head path
5      if GOAL-TEST(N) = TRUE
6          return reverse path
7      else newPaths ← empty list
8          for each M in MOVE-GEN(N)
9              newPaths ← (M : path,  g + k(N, M)) : newPaths
10         OPEN ← sort_g (newPaths ++ tail OPEN)
11 return empty list
```

BnB-2(S)

A variation of BnB, that prevents cyclic expansions like:
S, . . . , A, . . . , S, . . . , A, . . . ,

GENERALIZED-BnB

```
1  start with all possible solutions
2  repeat
3      refine the least (estimated cost) solution further
4  until the cheapest solution S is fully refined
5  return S
```