BUSCA ONLINE X OFFLINE

Até agora nos concentramos em agentes que utilizam algoritmos de busca offline: eles calculam uma solução completa antes de entrar no mundo real, e depois executam a solução sem recorrer a suas percepções. Em contraste, um agente de busca online, opera pela intercalação de planejamento e ação: primeiro ele executa uma ação, observa o ambiente e calcula a próxima ação.

Ideal para ambientes dinâmicos e/ou estocásticos! É aplicada a problemas de exploração, em que os estados e as ações são desconhecidos para os agente. Ex. robô implantado num edifício...

4.5 AGENTES DE BUSCA ONLINE:

Depois de cada ação, um agente online recebe uma percepção informando-o de qual estado ele alcançou; a partir dessa informação, ele pode ampliar seu mapa do ambiente.

Algoritmos online são bem diferentes dos offline: o A* tem a habilidade e expandir um nó em uma parte do espaço de estados e depois expandir imediatamente um nó em outra parte do espaço, por que expansão envolve simulação de ações, em vez de ações reais. Por outro lado, o algoritmo online só pode expandir o nó que ele ocupa fisicamente => **propriedade da localidade**!

A busca em profundidade tem esta propriedade!

Em consequência do retrocesso, a implementação a seguir só funcionará em espaços de estados nas quais as ações são reversíveis.

```
function ONLINE-DFS-AGENT(s') returns an action
  inputs: s', a percept that identifies the current state
  static: result, a table, indexed by action and state, initially empty
          unexplored, a table that lists, for each visited state, the actions not yet tried
          unbacktracked, a table that lists, for each visited state, the backtracks not yet tried
          s, a, the previous state and action, initially null

  if GOAL-TEST(s') then return stop
  if s' is a new state then unexplored[s'] ← ACTIONS(s')
  if s is not null then do
      result[a, s] ← s'
      add s to the front of unbacktracked[s']
  if unexplored[s'] is empty then
      if unbacktracked[s'] is empty then return stop
      else a ← an action b such that result[b, s'] = POP(unbacktracked[s'])
  else a ← POP(unexplored[s'])
  s ← s'
  return a
```

Figure 4.20 An online search agent that uses depth-first exploration. The agent is applicable only in bidirected search spaces.

4.5.1 BUSCA LOCAL ONLINE:

Assim como a busca em profundidade, a busca da subida na encosta tem a propriedade da localidade em suas expansões de nós. De fato, como ela mantém somente um estado corrente na memória, ele já é um algoritmo de busca online!

Problema: não é possível usar reinício aleatório => caminhamento aleatório *(random walking)*!!

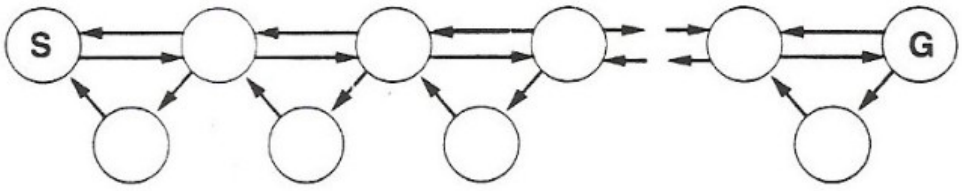

Figura 4.21 Um ambiente em que um percurso aleatório levará exponencialmente muitos passos para encontrar o objetivo.

4.5.2 LRTA* - LEARNING REAL TIME A*

LRTA* = subida da encosta com memória!

Idéia: armazenar a melhor estimativa atual H(s) do custo para alcançar o objetivo a partir de cada estado. H(s) começa sendo apenas a estimativa h(s) e é atualizada a medida que o agente ganha experiência.

Custo estimado para alcançar o objetivo através de s' = c(s,a,s') + H(s')

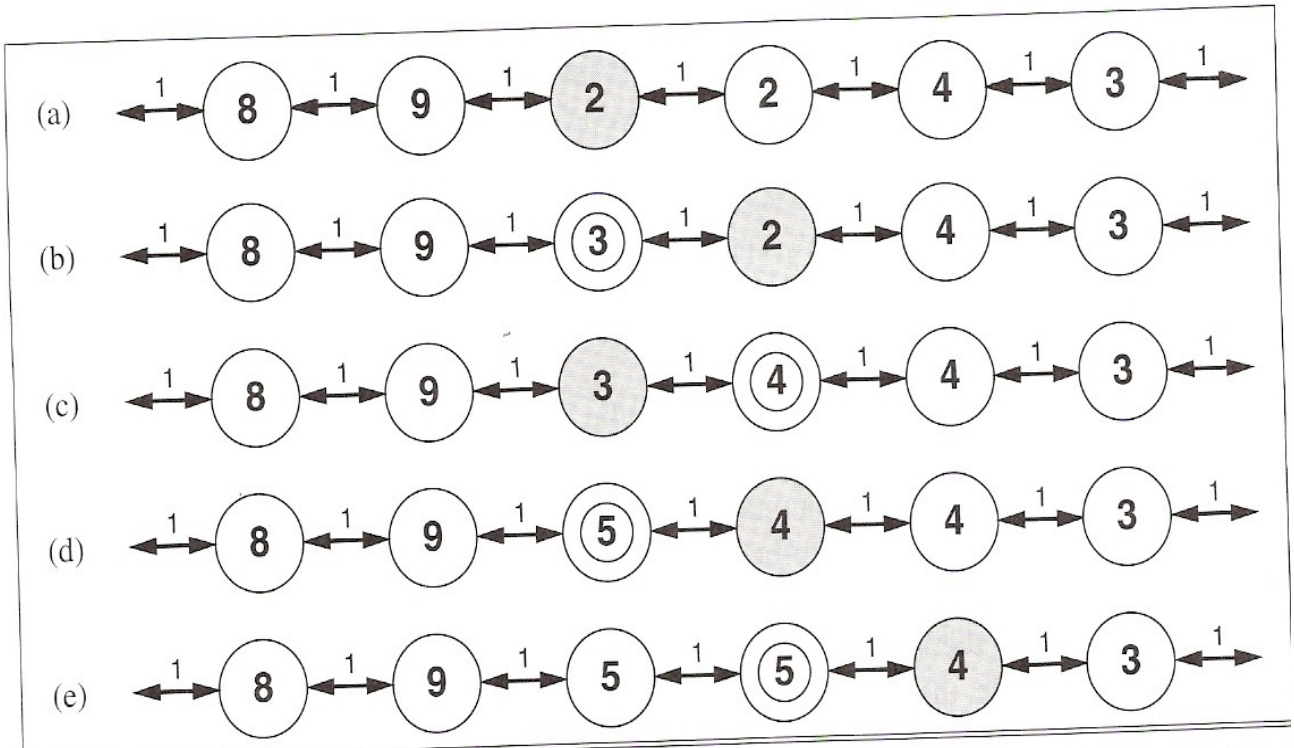

Figure 4.22 Five iterations of LRTA* on a one-dimensional state space. Each state is labeled with $H(s)$, the current cost estimate to reach a goal, and each arc is labeled with its step cost. The shaded state marks the location of the agent, and the updated values at each iteration are circled.