RACIOCÍNIO AUTOMÁTICO PARA PLANEAMENTO

Luís Morgado

2015

MÉTODOS DE PROCURA INFORMADA

- **Procura informada**
  - Estratégias de exploração do espaço de estados (controlo da procura) **tiram** partido de conhecimento do domínio do problema para **ordenar** a **fronteira de exploração**
  - **Avaliação de estado**
    - Estado é avaliado de acordo com o conhecimento do domínio do problema
      - **Custo** de **atingir o *estado*** a partir do estado inicial
        - » Custo do percurso realizado
      - **Custo** (**estimativa**) de **atingir a solução** a partir do *estado*
        - » Custo do percurso até à solução

PROCURA MELHOR-PRIMEIRO (*BEST-FIRST*)

```
function BEST-FIRST-SEARCH(problem, EVAL-FN) returns a solution sequence
  inputs: problem, a problem
          Eval-Fn, an evaluation function

  Queueing-Fn← a function that orders nodes by EVAL-FN
  return GENERAL-SEARCH(problem, Queueing-Fn)
```

Figure 4.1 An implementation of best-first search using the general search algorithm.

[Russel & Norvig, 1995]

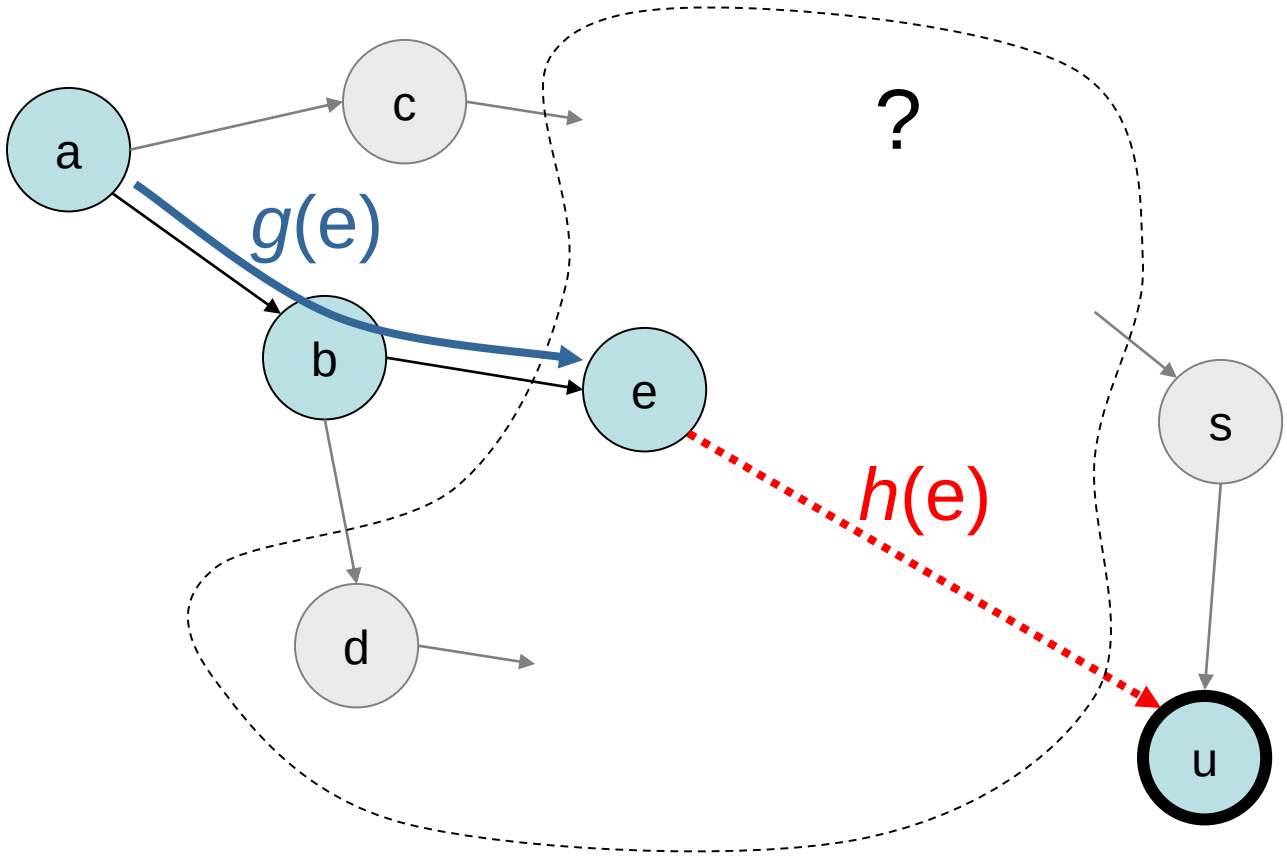
Função de Avaliação *f*(*n*)

- $f(n) = g(n) + h(n)$
  - $g(n)$: Custo do percurso desde o nó inicial até ao nó n
  - $h(n)$: Estimativa do custo do percurso desde o nó n até ao nó objectivo
  - $f(n)$: Estimativa do custo do percurso total (solução) através do nó n
- $h(n)$ é tipicamente uma estimativa
  - **Função *heurística***

FUNÇÃO HEURÍSTICA *h*(*n*)

- Representa uma estimativa do custo do percurso desde o nó *n* até ao nó objectivo
- Reflecte conhecimento acerca do domínio do problema
- O seu valor é independente do percurso até *n*
  - Depende apenas de:
    - **Estado** associado a *n*
    - **Objectivo**

FUNÇÃO DE AVALIAÇÃO *f*(*n*)



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

PROCURA MELHOR-PRIMEIRO (*BEST-FIRST*)

3 variantes principais

- $f(n) = g(n)$
  - **Procura de Custo Uniforme**
    - Não tira partido de conhecimento do domínio do problema expresso através da função $h(n)$
- $f(n) = h(n)$
  - **Procura Sôfrega (*Greedy Search*)**
    - Não tem em conta o custo do percurso explorado
    - Minimização de custo **local**
- $f(n) = g(n) + h(n)$
  - **Procura A*** (heurística admissível)
    - Minimização de custo **global**

PROCURA SÔFREGA (*GREEDY SEARCH*)

```
function GREEDY-SEARCH( problem) returns a solution or failure
    return BEST-FIRST-SEARCH( problem, h)
```

[Russel & Norvig, 1995]

Função de avaliação de estado *f*(*s*):

- Custo estimado a partir de ***s*** até ao objectivo

$f(s) = h(s)$

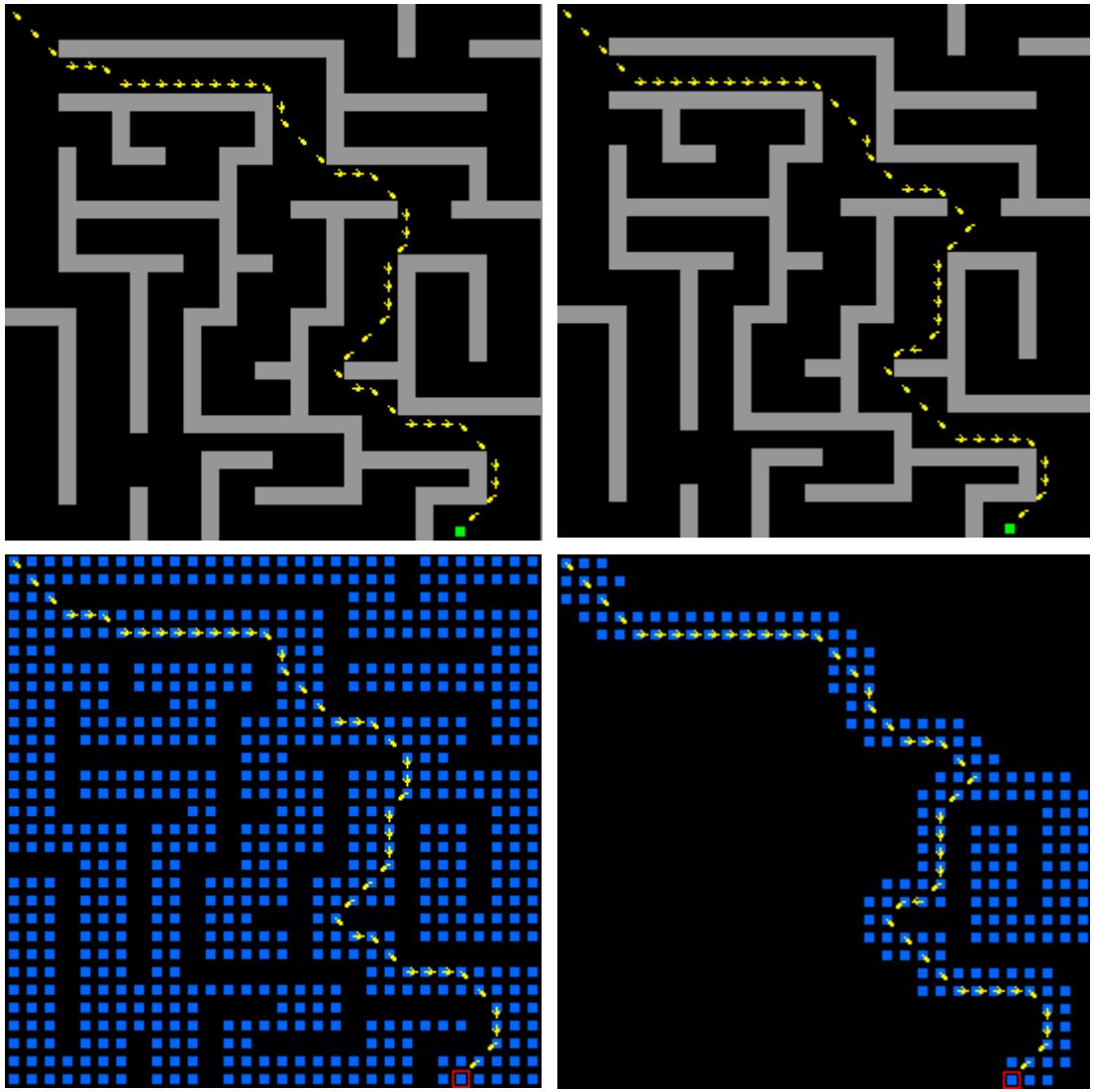
ANÁLISE LOCAL

Minimização de custo local até ao objectivo

Soluções sub-óptimas

PROCURA MELHOR-PRIMEIRO (*BEST-FIRST*)

Procura de custo uniforme (solução óptima)



Procura sôfrega (solução sub-óptima)

PROCURA A*

```
function A*-SEARCH(problem) returns a solution or failure
    return BEST-FIRST-SEARCH( problem,g + h)
```

[Russel & Norvig, 1995]

Função de avaliação de estado *f*(*s*):

- Custo estimado da solução através do estado ***s***

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

ANÁLISE GLOBAL

Minimização do custo global até ao objectivo

Soluções óptimas

PROCURA A*

- **Heurística admissível**
  - $0 \leq h(n) \leq h^*(n)$
  - $h^*(n)$
    - Custo efectivo mínimo do nó *n* até ao objectivo (percurso óptimo)
- Uma **heurística admissível** é ***optimista***
  - A estimativa de custo é sempre inferior ou igual ao custo efectivo mínimo
  - Para um nó objectivo n_{obj}
    - $h(n_{obj}) = 0$

PROCURA A*

- C^* - Custo da solução óptima
- n - Nó na fronteira de exploração

  $f(n) = g(n) + h(n) \leq C^*$ (se $h(n)$ admissível)
- m - Nó sub-óptimo na fronteira de exploração

  $f(m) = g(m) + h(m)$
- Se m for um nó objectivo

  $h(m) = 0$

  $f(m) = g(m) > C^*$
- Então

  $f(n) \leq C^* < f(m)$

  m não será expandido e a **solução encontrada será óptima**

PROCURA A*

- Método de procura de **eficiência óptima**
  - Nenhum outro algoritmo expandirá menos nós, mantendo as características de ser **completo** e **óptimo**, excepto nas situações de escolha entre nós com $f(n) = C^*$
- **No entanto, não resolve o problema da complexidade combinatória**
  - O número de nós expandidos dentro do contorno do nó objectivo contínua a ser uma **função exponencial** da dimensão do percurso até ao objectivo
  - Função heurística afecta o contorno de procura
    - Pode não ser suficiente

COMPARAÇÃO DE MÉTODOS DE PROCURA

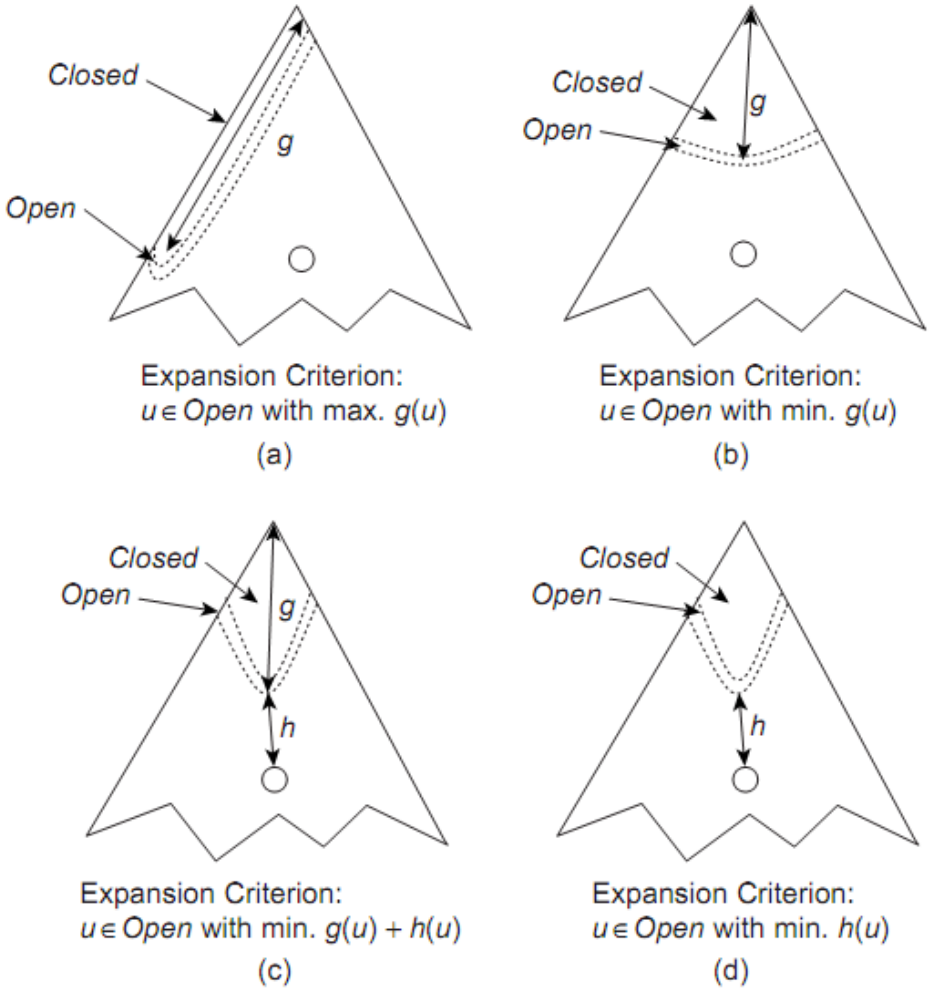
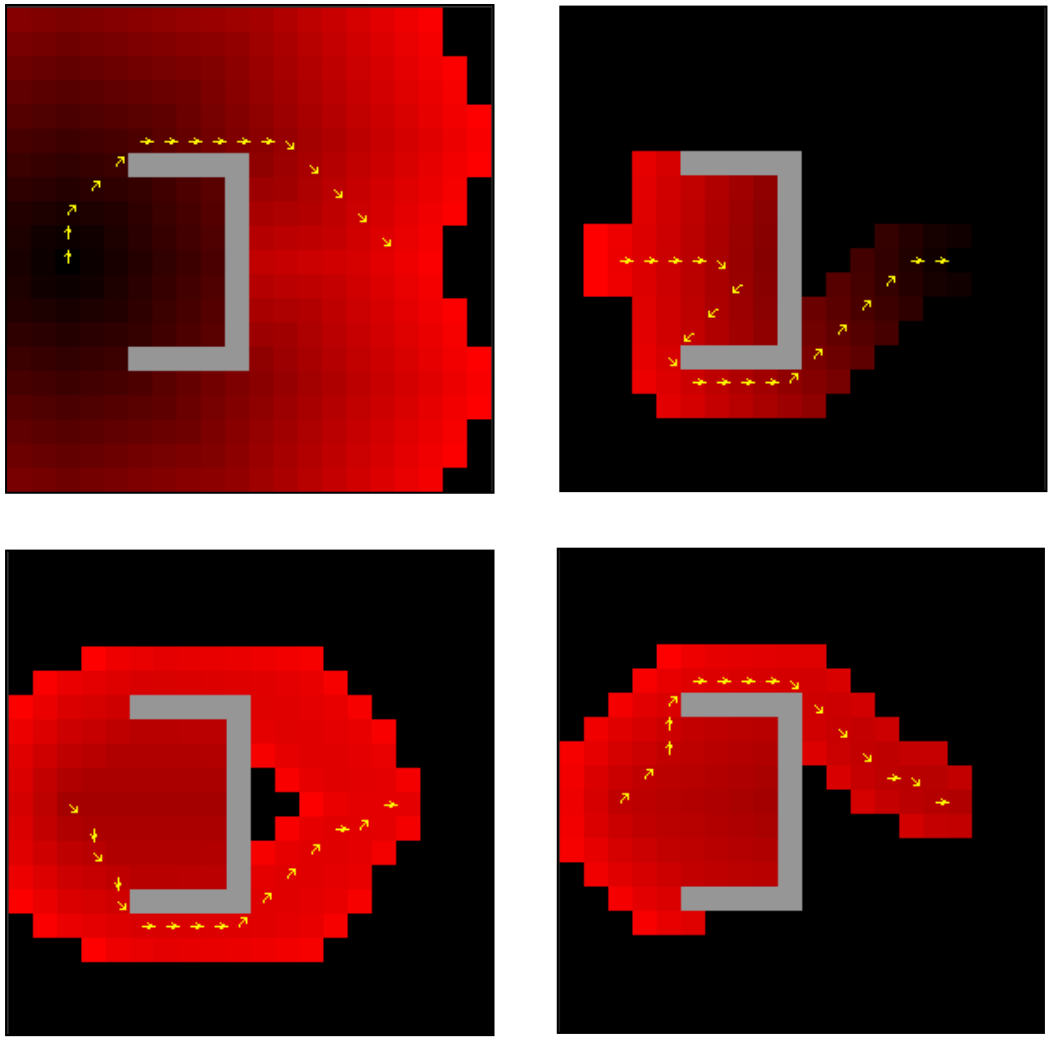


FIGURE 2.17

Different search strategies: (a) DFS, (b) BFS, (c) A*, and (d) greedy best-first search.

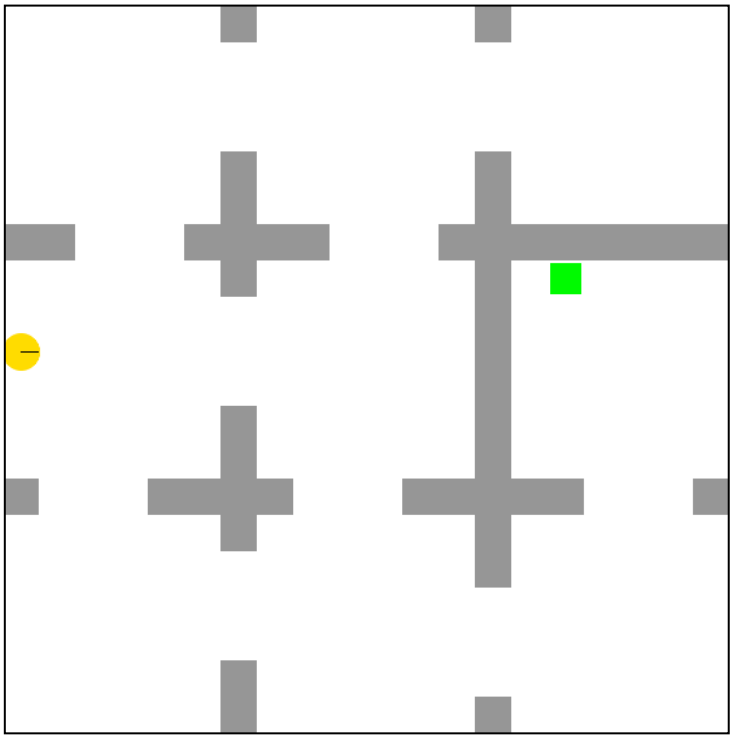
[Edelkamp & Schrodl, 2012]

PROCURA EM ESPAÇOS DE ESTADOS
COMPARAÇÃO DE MÉTODOS DE PROCURA



PLANEAMENTO PARA NAVEGAÇÃO AUTÓNOMA

Planeamento de trajectos de um veículo autónomo



ALGORITMO *WAVEFRONT* (*FRENTE-ONDA*)

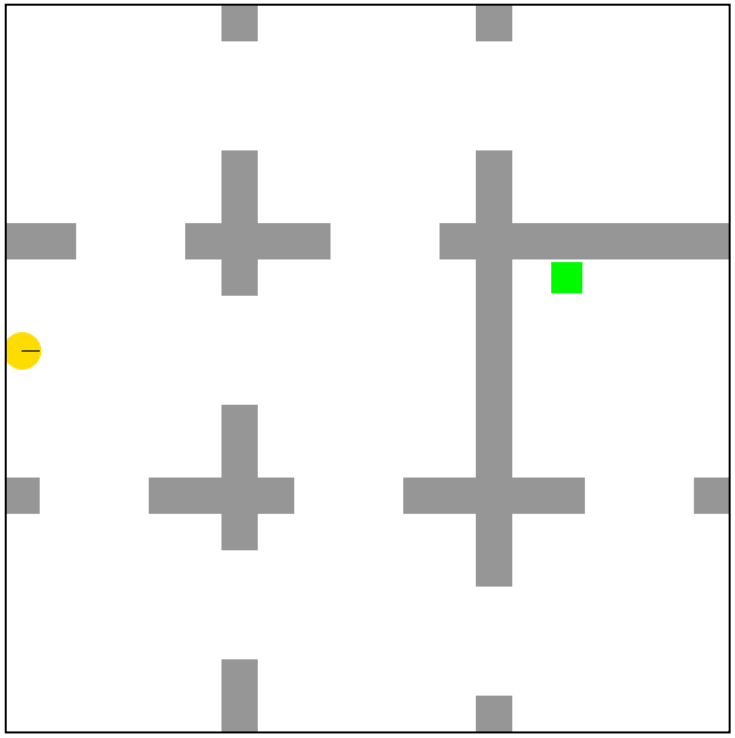
- Abordagem baseada em campos de valor
  - **Funções de valor unimodais**
    - Um único máximo (global)
  - **Campos de valor internos (motivacionais)**
    - **Simulação do mundo** com base em representações internas (**modelo do mundo**)
    - Planeamento modelado como navegação num campo de valor
      - Procura local (e.g. *hill-climbing*)
      - Estratégia global (e.g. política comportamental)

ALGORITMO *WAVEFRONT* (*FRENTE-ONDA*)

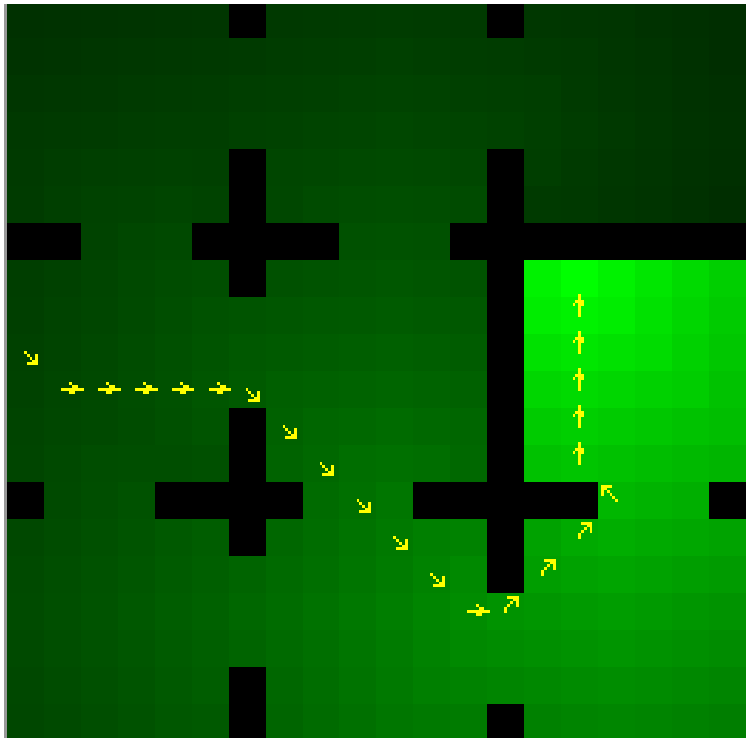
GERAR CAMPO MOTIVACIONAL INTERNO

A partir da representação (modelo) do mundo

- Estrutura
- Dinâmica (bidireccional)
- Valor - $V : S \rightarrow \Re$



Óptimos globais



REPRESENTAÇÃO DO MUNDO

$\boldsymbol{S}$ – Estados

- Conjunto de estados do mundo

$\boldsymbol{A}$ – Acções

- Conjunto de acções possíveis

$\boldsymbol{T : S \times A \rightarrow S}$ – Função de transição de estado

- Transições de estado

$\boldsymbol{Objectivos \subset S}$

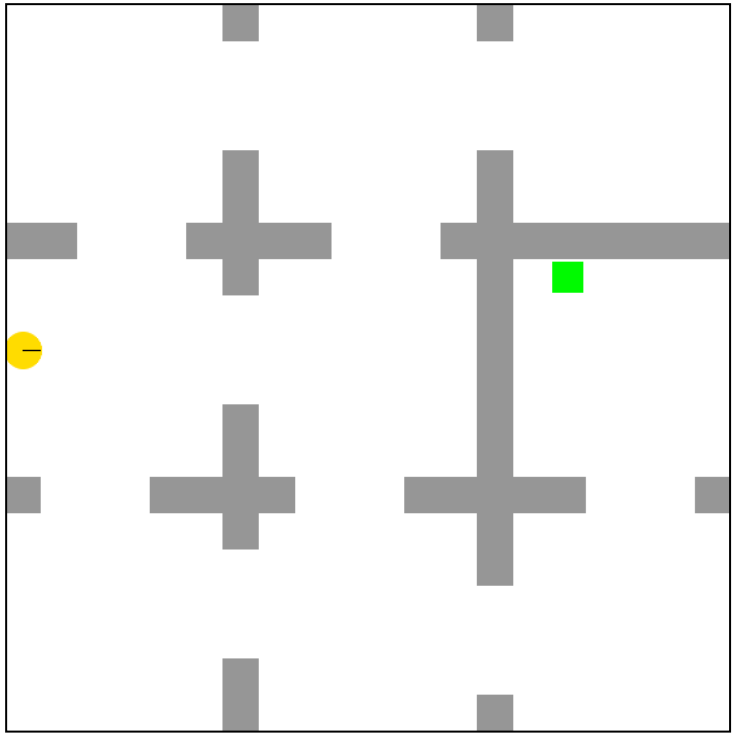
- Conjunto de objectivos do sistema

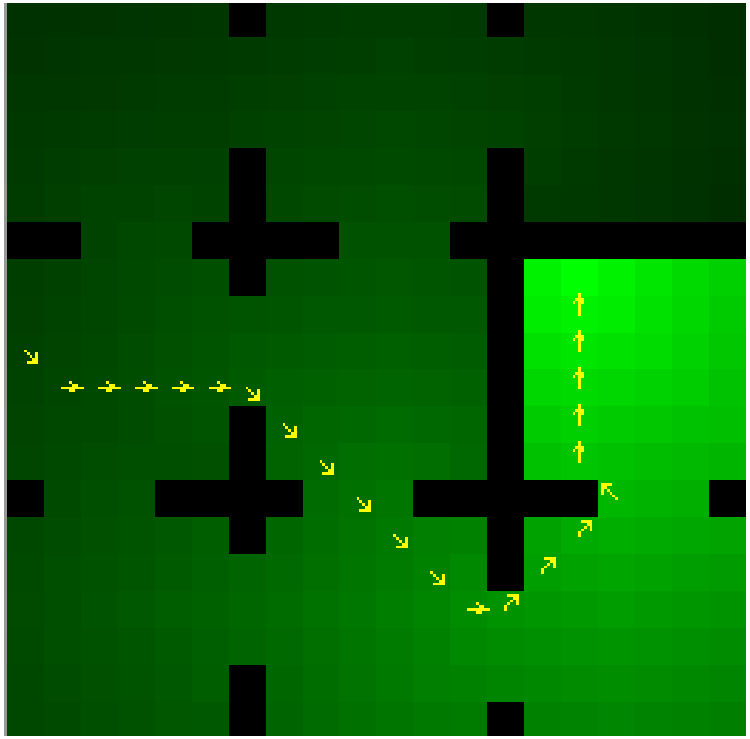
ALGORITMO *WAVEFRONT (FRENTE-ONDA)*

```
PROPAGAR-VALOR(objectivos)
1.  V = {}
2.  frente-de-onda = []
3.  for s in objectivos:
4.      V[s] = VALOR_MAX
5.      frente-de-onda.append(s)
6.  while frente-de-onda:
7.      s = frente-de-onda.pop(0)
8.      for s' in adjacentes(s):
9.          v = V[s] * γ dist(s, s')
10.         if v > V.get(s', -∞):
11.           V[s'] = v
12.           frente-de-onda.append(s')
```

PLANO DE ACÇÃO

Exemplo: algoritmo *hill-climbing*





POLÍTICA COMPORTAMENTAL

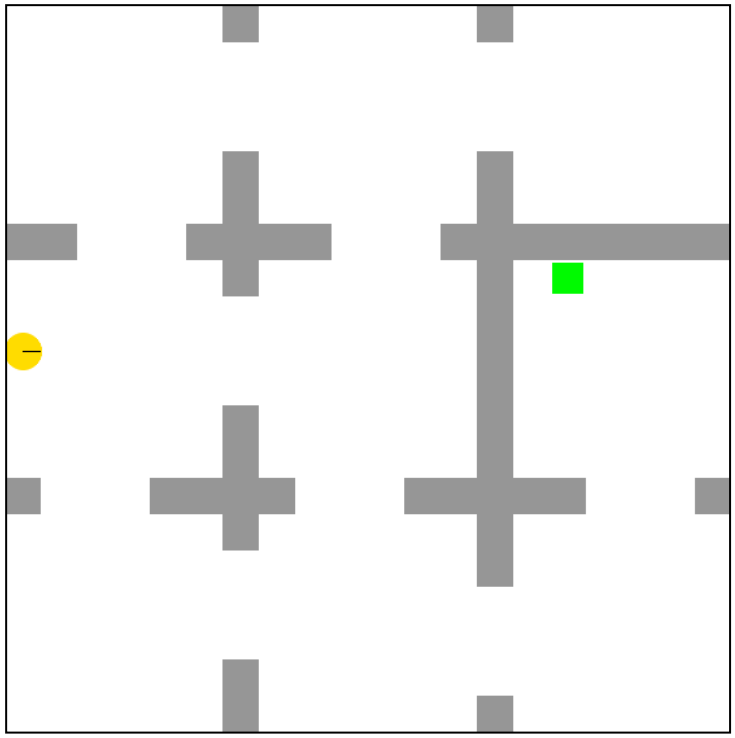
- Forma de representação do comportamento do agente
- Define qual a acção que deve ser realizada em cada estado (estratégia de acção)
- S – conjunto de estados do mundo
- $A(s)$ – conjunto de acções possíveis no estado $s \in S$
- Política **determinista**

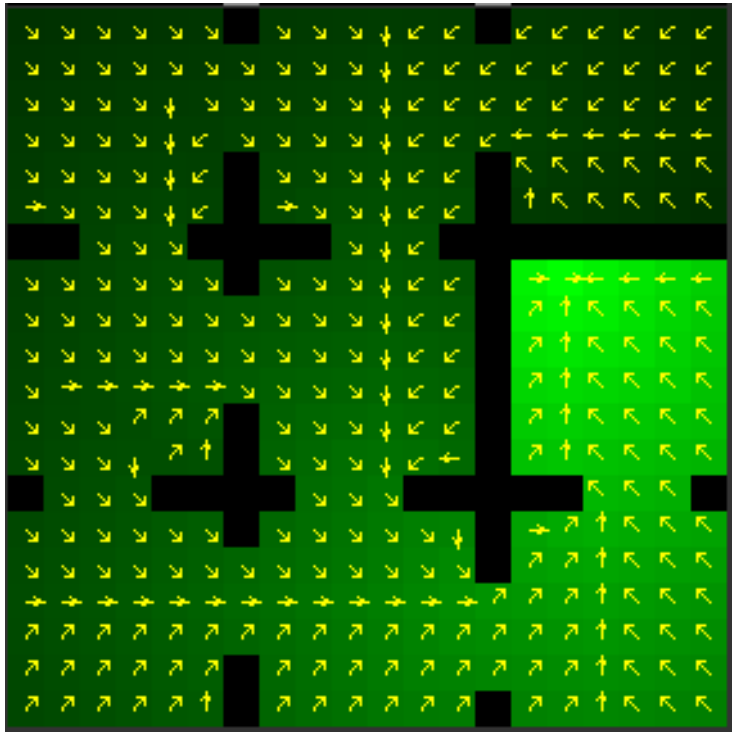
$$\pi : S \rightarrow A(s) \; ; \; s \in S$$

POLÍTICA COMPORTAMENTAL

Exemplo: política determinista

$$\pi : S \rightarrow A(s) \; ; \; s \in S$$





ALGORITMO *WAVEFRONT (FRENTE-ONDA)*

- Problemas
  - Utilização extensiva de memória
    - Aplicável apenas a regiões limitadas do espaço de estados
  - Requer operadores bidireccionais

BIBLIOGRAFIA

[Russel & Norvig, 1995]

S. Russell, P. Norvig, *Artificial Intelligence: A Modern Approach*, 1st Edition, Prentice Hall, 1995

[Russel & Norvig, 2010]

S. Russell, P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd Edition, Prentice Hall, 2010

[Nilsson, 1998]

N. Nilsson , *Artificial Intelligence: A New Synthesis*, Morgan Kaufmann 1998

[Luger, 2009]

G. Luger , *Artificial Intelligence: Structures and Strategies for Complex Problem Solving* , Addison-Wesley, 2009

[Jaeger & Hamprecht, 2010]

M. Jaeger, F. Hamprecht, *Automatic Process Control for Laser Welding*, Heidelberg Collaboratory for Image Processing (HCI) , 2000

[Pearl, 1984]

J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley, 1984

[Montemerlo, 2008]

M. Montemerlo *et al.*, *Junior: The Stanford Entry in the Urban Challenge*, Stanford Artificial Intelligence Lab, 2008