CES -161 - Modelos Probabilísticos em Grafos

Markov Decision Process

Prof. Paulo André Castro
pauloac@ita.br
www.comp.ita.br/~pauloac
Sala 110,
IEC-ITA

Aprendizado - paradigmas

- Aprendizado supervisionado
  - O crítico comunica a EA o erro relativo entre a ação que deve ser tomada idealmente pelo EE e a ação efetivamente escolhida pelo agente. Pares (corretos) de entrada/saída podem ser observados (ou demonstrados por um supervisor).
- Aprendizado por reforço
  - O crítico comunica apenas uma indicação de desempenho (indicação de quão bom ou ruim é o estado resultante), por vezes de modo intermitente e apenas quando situações dramáticas são atingidas (*feedback* indireto, com retardo).
- Aprendizado não-supervisionado
  - O crítico não envia nenhum tipo de informação ao EA, não há "pistas" sobre as saídas corretas (geralmente utiliza-se regularidades, propriedades estatísticas dos dados sensoriais)
    - Busca-se encontrar padrões ou estruturas / agrupamentos nos dados. Inclui por exemplo técnicas de clusterização

Decisões Sequenciais em Ambientes Estocásticos

- O resultado imediato (próximo estado) não depende apenas do estado atual e da ação do agente, outros fatores influenciam de modo não plenamente conhecido (estocástico)

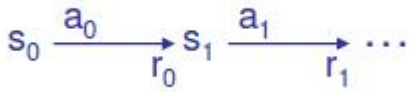
- O estado atual e a ação tomada definem um conjunto de possíveis estados sucessores com as respectivas probabilidades

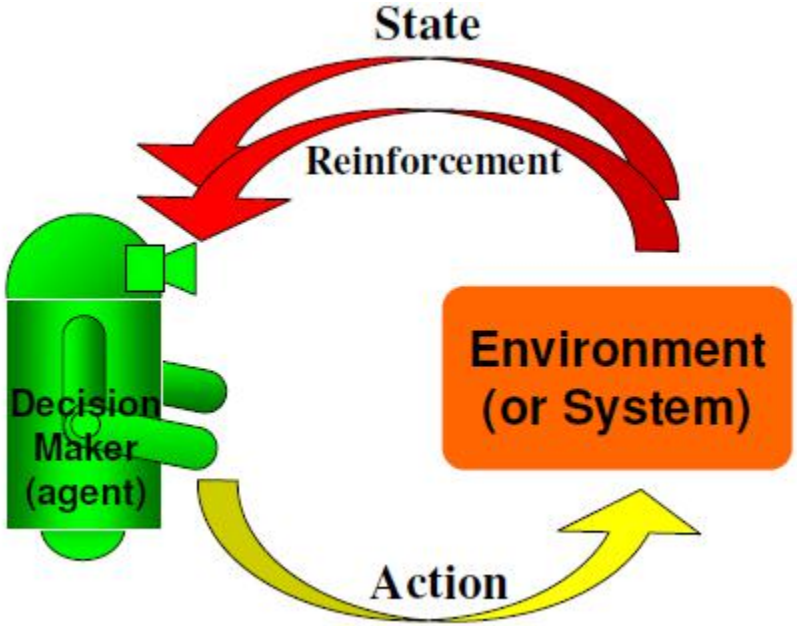
- O agente tem como objetivo maximizar seu retorno acumulado a longo prazo
  - Decidir por um caminho ruim agora é plenamente aceitável se no futuro houver recompensa significativa…

Sequential Decisions

- The sequential decision problem can be seen as:

1. Observe system's state
2. Pick and execute an Actioin
   (Systems evolves to a new state)
3. Observe an immediate reinforc
4. Repeat steps 1 – 3





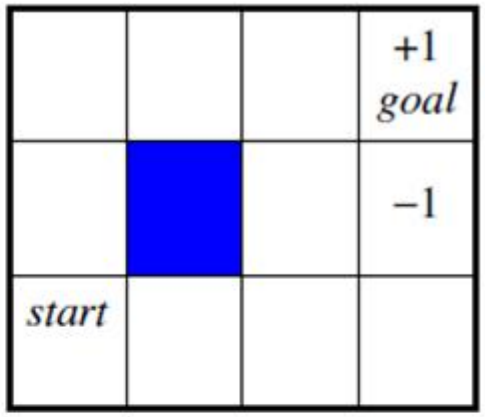
- (We implicitly assumed discrete time!)

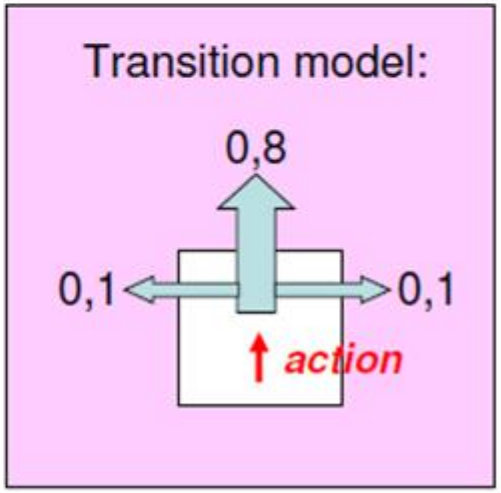
Example

- 4×3 discrete fully-observed environment
- Actions: Up, Down, Left and Right
- Sequence [U, U, R, R, R]:

  (i) goes up around the barrier and reaches the goal state (4,3) with probability $0{,}8^5 = 0.32768$.

  (ii) there is also a chance of accidentally reaching the goal by going $(1,1) \rightarrow (2,1) \rightarrow (3,1) \rightarrow (3,2) \rightarrow (3,3) \rightarrow (4,3)$ with probability $0{,}1^4 \times 0{,}8$. Total = 0.32776





Example - 2

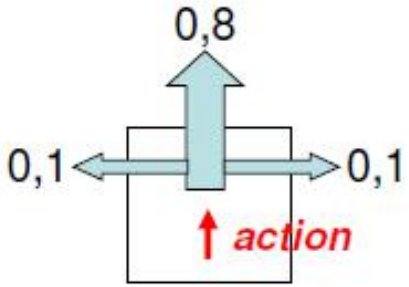
- **Utility function** for the agent depends on a sequence of states (environment ***history***)
- In each state **s**, the agent receives a **reinforcement r(s)**, which may be positive or negative, but must be **bounded.**
- Utility = sum of the rewards received
- Here: $\mathbf{r(s) = -0.04}$ $\forall s$ except $r(4,3) = +1$ and $r(4,2) = -1$

➔ reach goal after 10 steps:

$$\text{utility} = 10 \times -0{,}04 + 1 = 0{,}6$$

-0,04	-0,04	-0,04	+1 *goal*
-0,04		-0,04	−1
-0,04 *start*	-0,04	-0,04	-0,04

Transition model:



Processo Decisório de Markov (Markov Decision Process)

- Método de decisão para problema de decisão sequencial pode ser modelado como um modelo de transição Markoviano e reforços aditivos

- O qualificador Markov significa que as transições dependem de um subconjunto dos últimos estados e da ação selecionada.

Notation: $X_{a:b} = X_a; X_{a+1}; \ldots; X_{b-1}; X_b$

Markov assumption: X_t depends on bounded subset of $X_{0:t-1}$

- First-order Markov process: $\mathbf{p(X_t|X_{0:t-1}) = p(X_t|X_{t-1})}$
- Second-order Markov process: $\mathbf{p(X_t|X_{0:t-1}) = p(X_t|X_{t-2},X_{t-1})}$

Formal Definition of a MDP

An MDP is defined as $\langle S, A, p, r \rangle$:

- **S** is the set of possible system states (arbitrary finite set);
- **A** is the set of allowable actions (arbitrary finite set);
- **p**: $S \times A \times S \rightarrow [0,1]$ is the **transition probability function**;
- **r**: $S \times A \rightarrow \Re$ is the **reinforcement function**;

Formal definition of a MDP - 2

- The set **A** of allowable actions:
  - $A = \cup_{s \in S} A_s$ where A_s is the set of allowable actions in state $s \in S$
  - Or we might restrict the model: $A = A_s$ for all $s \in S$
- The transition probability function **p:**
  - $p(j \mid s, a)$ – or $p(s, a, j)$ – denotes the probability that the system is in state $j \in S$ at time t+1, when the decision maker performs action $a \in A_s$ in state $s \in S$ at time t.
  - $\sum_{j \in S} p(j \mid s, a) = 1$

Formal definition of a MDP - 3

- The reinforcement function **r:**
  - r(s,a), denotes the value of the reward or cost received when performing $a \in A_s$ in $s \in S$ at time t.
  - When positive, r(s,a) is an *income*, and when negative a *cost*.
  - Can be:

    r(s);

    r(s,a);

    r(s,a,j) with $r(s,a) = \sum_{j \in S} r(s,a,j)\, p(j \mid s, a)$

Exemplo 1: Controle de Inventário

- **Problema: comprar uma quantidade de um certo produto a intervalos regulares (em um total de N intervalos) de modo a satisfazer uma certa demanda**
- Estado: s_k = estoque no começo do período k
- Ação: a_k = compra feita no começo do período k
- Uma perturbação aleatória w_k = demanda no período k, respeitando uma certa distribuição de probabilidade
- Reforço $r_k = r(s_k) + ca_k$, onde $r(s_k)$ é o custo de estocar s_k unidades do produto no período k e c é o custo unitário do produto comprado.

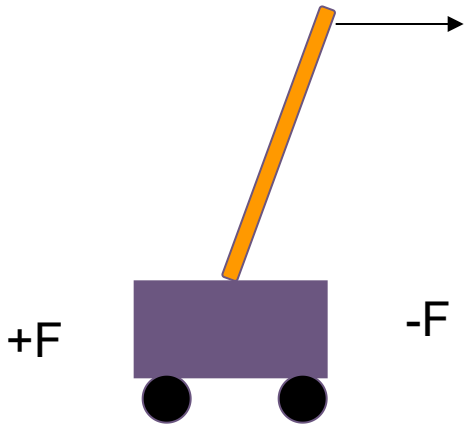
Exemplo 1: Controle de Inventário

- Evolução do estado: $s_{k+1} = s_k + a_k - w_k$
- Função de custo a ser minimizada:

$$V(s_o) = E\left\{ r(s_N) + \sum_{k=0}^{N-1} \left(r(s_k) + c a_k \right) \right\}$$

Exemplo 2: Pêndulo Invertido

- Problema: controlar um pêndulo invertido exercendo forças +F ou -F sobre a base do carrinho (controle *bang-bang*). "Controlar" significa não permitir que a barra caia



Exemplo 2: Pêndulo Invertido

- Estado: quádrupla $(x_t, \partial x_t, \theta_t, \partial \theta_t)$
- Ação a_k : +F ou -F
- Reforço: -1 em caso de falha, senão 0.
- Evolução do estado: $s_{k+1} = f(s_k, a_k)$ (?)
- Possível função recompensa a ser maximizada:

$$V(s_o) = \mathrm{E}\left\{\sum_{t=0}^{\infty} \gamma^k r_t\right\}$$

desconto temporal $\gamma < 1$: POR QUÊ?

O que é uma solução para um Problema de Markov?

- Uma sequencia de ações (plano) pode resolver um ambiente Estocástico?

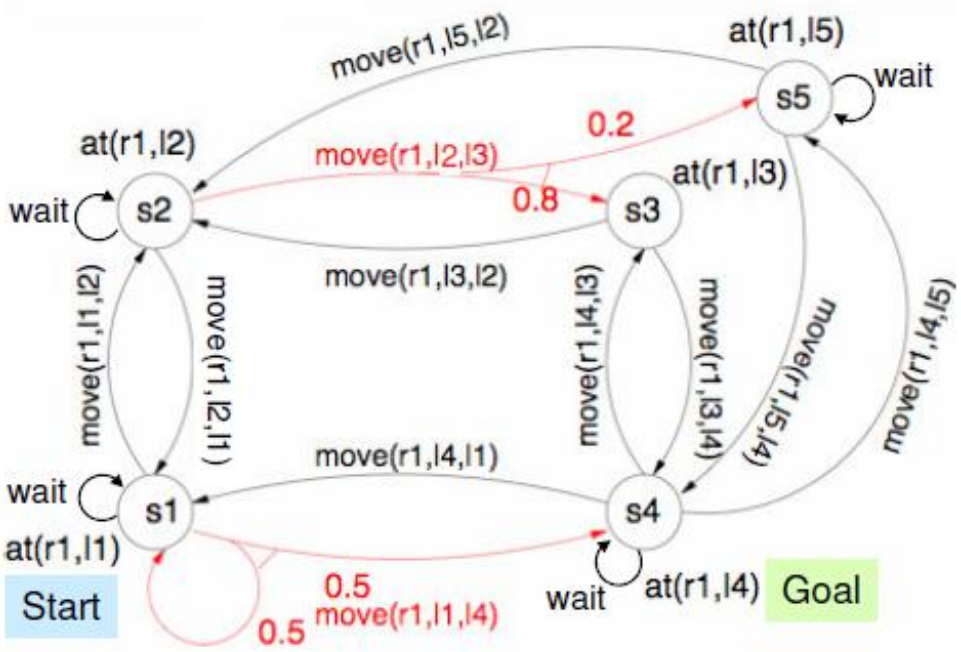
- Políticas (ou Estratégia) versus Planos…Formalização

O que é uma solução para um Problema de Markov?

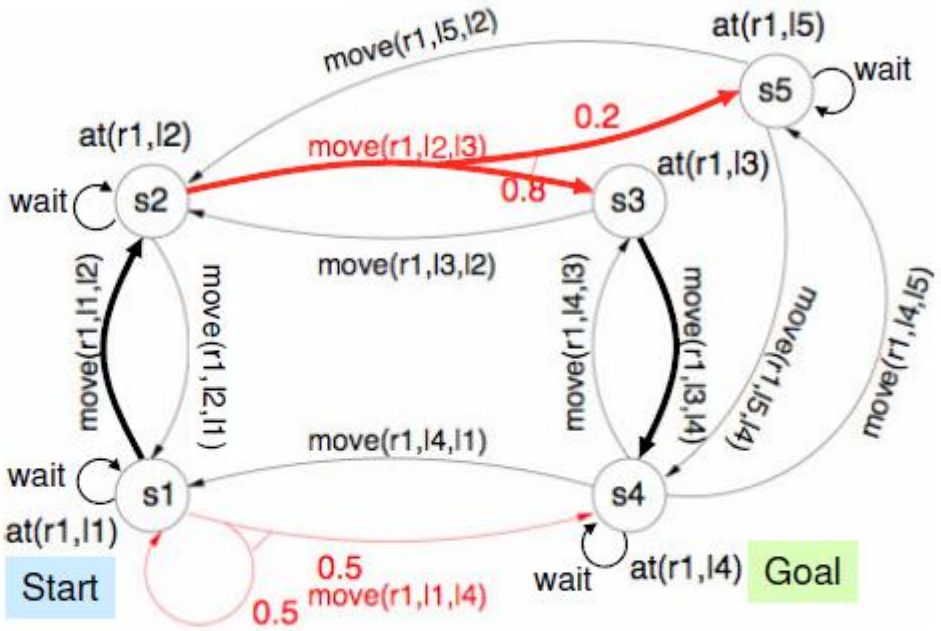
- Exemplo: Controle de Movimentação de robô modelado como um PDM
  - No início, robô está em l1 (estado s_1)
  - Objetivo é levar o robô até l4 (estado s_4)
  - Robô pode "deslizar" ao tentar se mover de uma posição para outra

MDP Example – Mobile Robot

- Robot r1 starts at location l1
  - State s1 in the diagram
- Objective is to get r1 to location l4
  - State s4 in the diagram

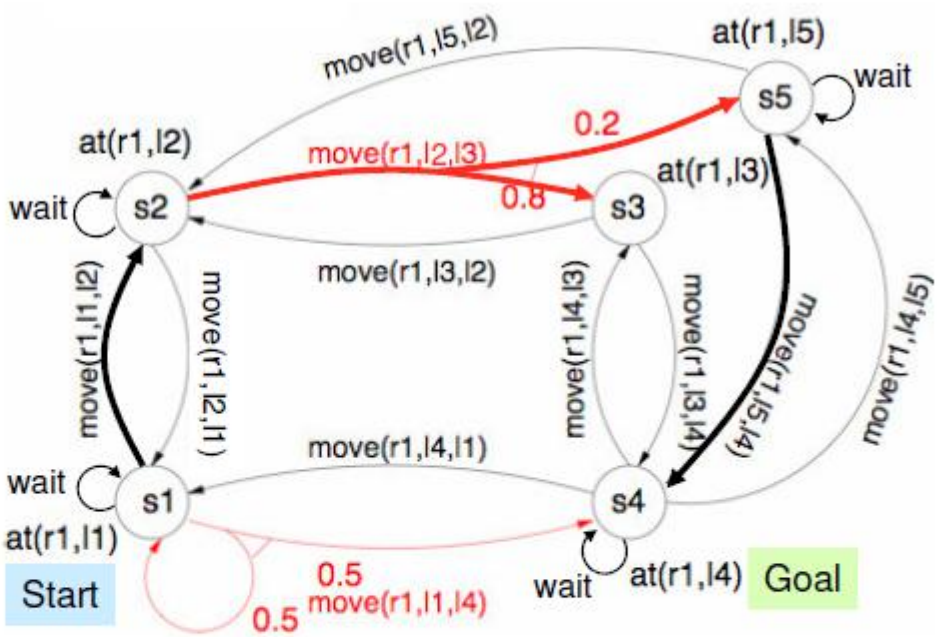


MDP Robot – 2



- e.g.,
  - » ⟨move(r1,l1,l2), move(r1,l2,l3), move(r1,l3,l4)⟩

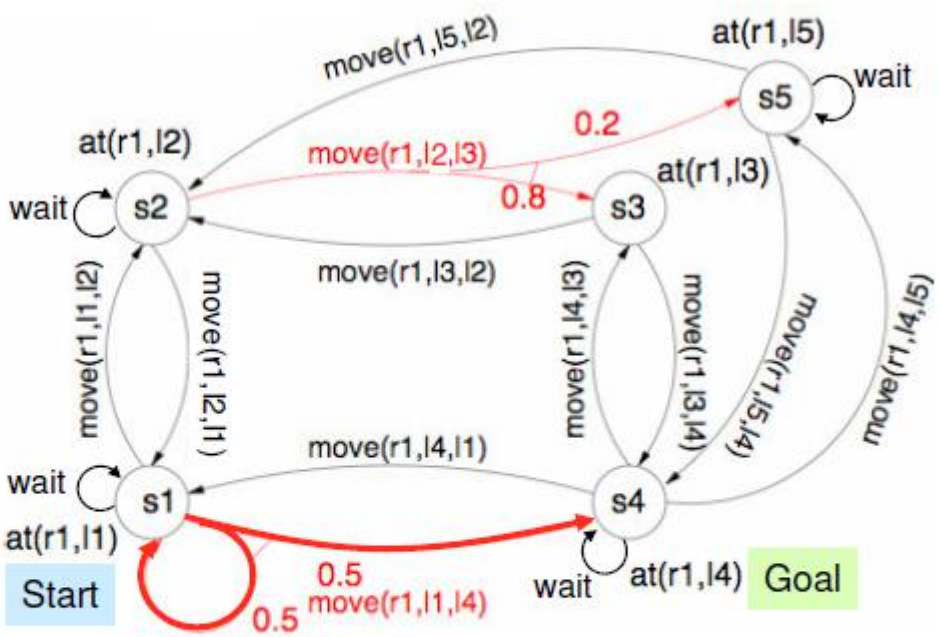
MDP Robot – 3



- e.g.,

  ⟨move(r1,l1,l2), move(r1,l2,l3), move(r1,l5,l4)⟩

MDP Robot – 4



- No fixed sequence of actions can be a solution, because we can't guarantee we'll be in a state where the next action is applicable
  - e.g.,

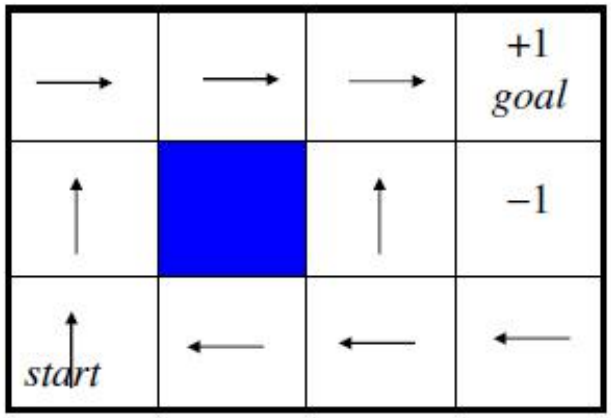
    ⟨move(r1,l1,l4)⟩

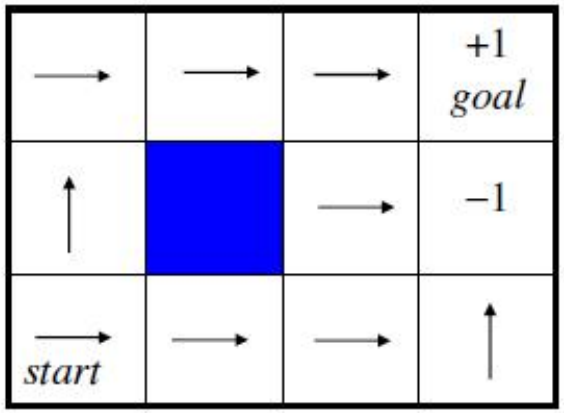
Seqüencia não funciona...

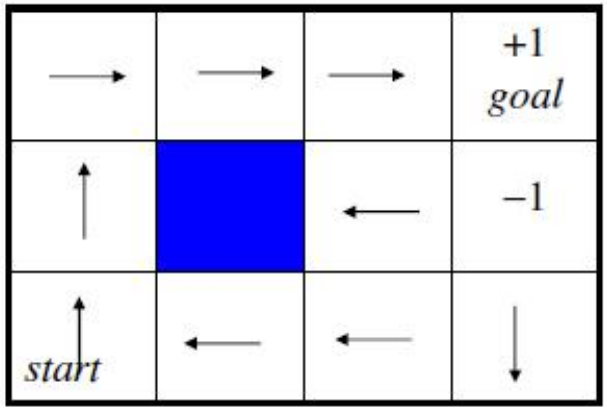
- É necessário uma função que mapeie estados a ações. Esta função é chamada de estratégia ou politica(policy)

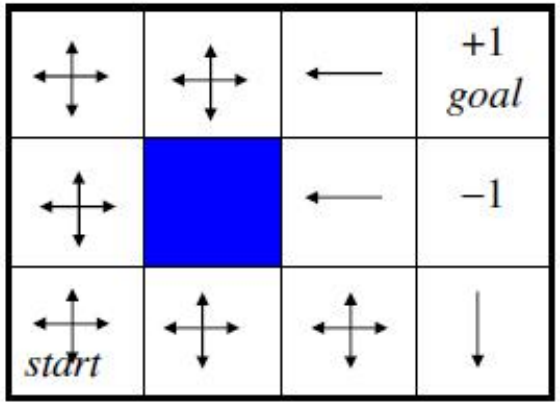
$$\Pi: \mathrm{S} \rightarrow \mathrm{A}, \quad \pi(s) = \mathrm{a} \in \mathrm{A}$$

Policies for the Grid World







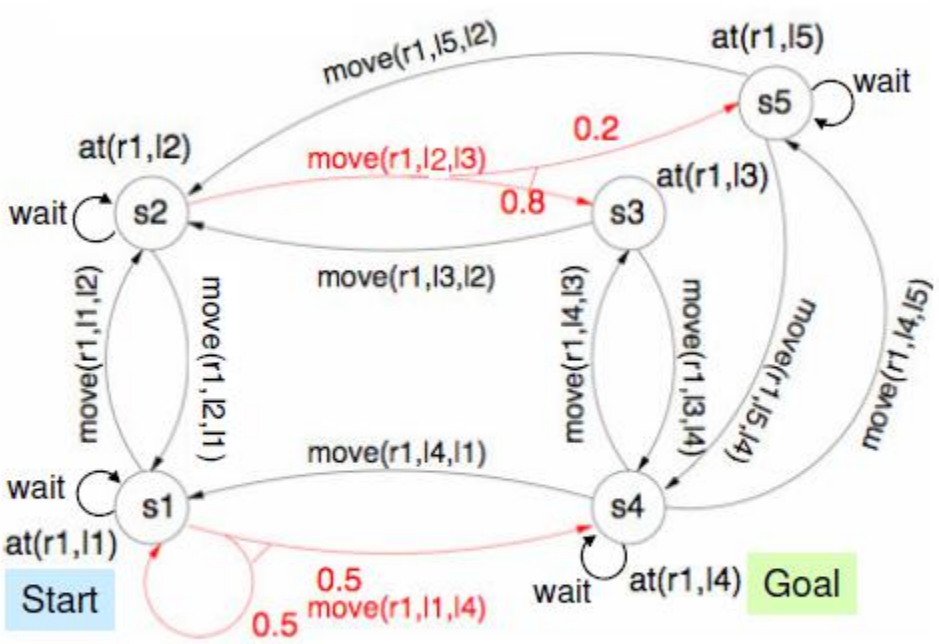


Exemplos de Políticas – Problema 2

π_1 = {(s1, move(r1,l1,l2)),
(s2, move(r1,l2,l3)),
(s3, move(r1,l3,l4)),
(s4, wait),
(s5, wait)}

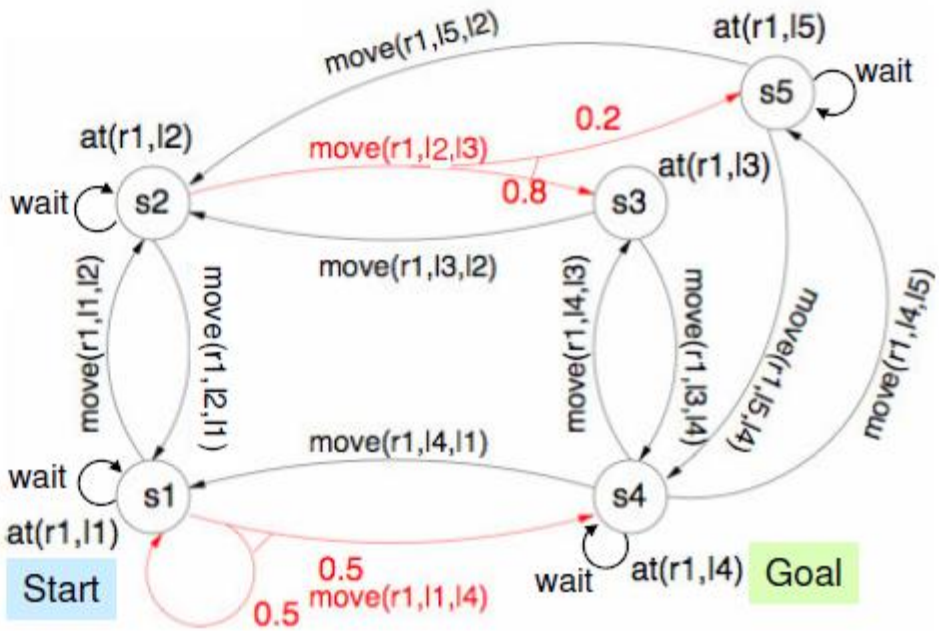
π_2 = {(s1, move(r1,l1,l2)),
(s2, move(r1,l2,l3)),
(s3, move(r1,l3,l4)),
(s4, wait),
(s5, move(r1,l5,l4))}

π_3 = {(s1, move(r1,l1,l4)),
(s2, move(r1,l2,l1)),
(s3, move(r1,l3,l4)),
(s4, wait),
(s5, move(r1,l5,l4)}



Initial state

- It is possible to define a probability distribution over states for the first, but for simplicity. Let's define s_0



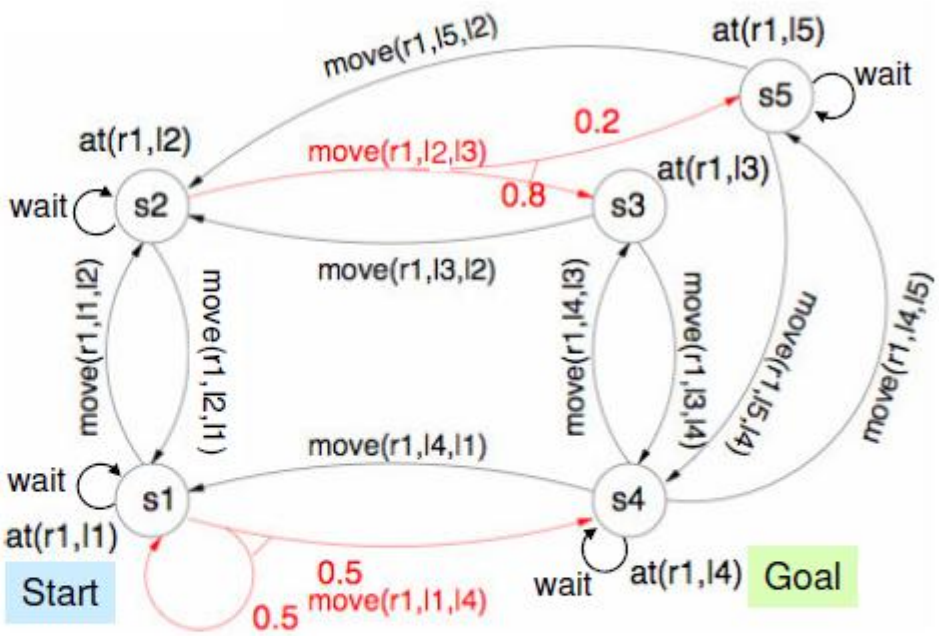
  - » $P(s_0) = 1$
  - » $P(s_i) = 0$ for $i \neq 0$
- In the example, $P(s_1) = 1$, and $P(s) = 0$ for all other states

History: sequence of states

- *History*: sequence of system states

$h = \langle s_0, s_1, s_2, s_3, s_4, \ldots \rangle$

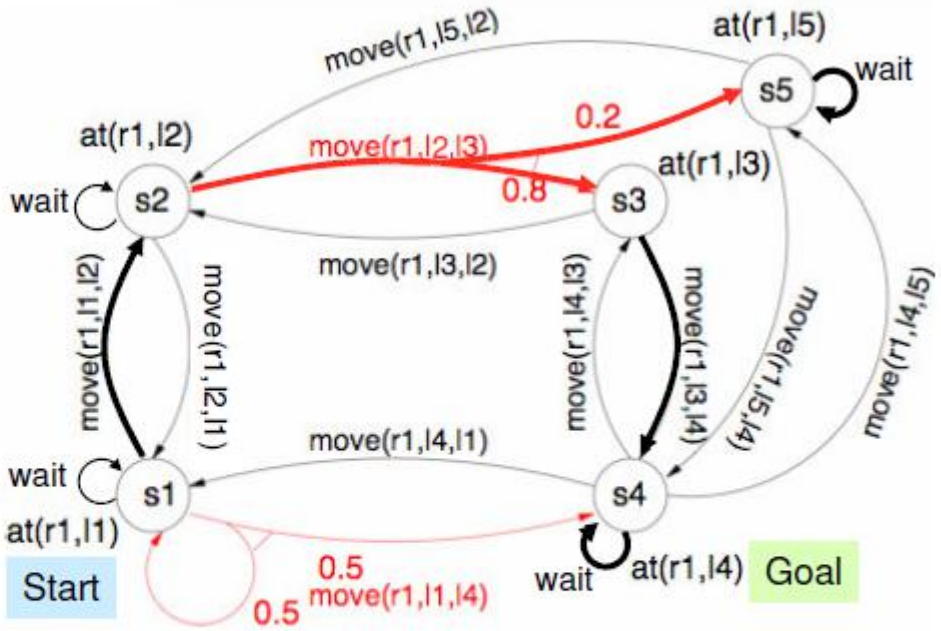
$h_0 = \langle$s1, s2, s1, s2, s1, … $\rangle$
$h_1 = \langle$s1, s2, s3, s4, s4, … $\rangle$
$h_2 = \langle$s1, s2, s5, s5, s5, … $\rangle$
$h_3 = \langle$s1, s2, s5, s4, s4, … $\rangle$
$h_4 = \langle$s1, s4, s4, s4, s4, … $\rangle$
$h_5 = \langle$s1, s1, s4, s4, s4, … $\rangle$
$h_6 = \langle$s1, s1, s1, s4, s4, … $\rangle$
$h_7 = \langle$s1, s1, s1, s1, s1, … $\rangle$



- **Each policy induces a probability distribution over histories**
  - If $h = \langle s_0, s_1, \ldots \rangle$ then $P(h \mid \pi) = P(s_0) \prod_{i \geq 0} p_{\pi(s_i)}(s_{i+1} \mid s_i, a_i)$

History: sequence of states – 2

π_1 = {(s1, move(r1,l1,l2)),
(s2, move(r1,l2,l3)),
(s3, move(r1,l3,l4)),
(s4, wait),
(s5, wait)}



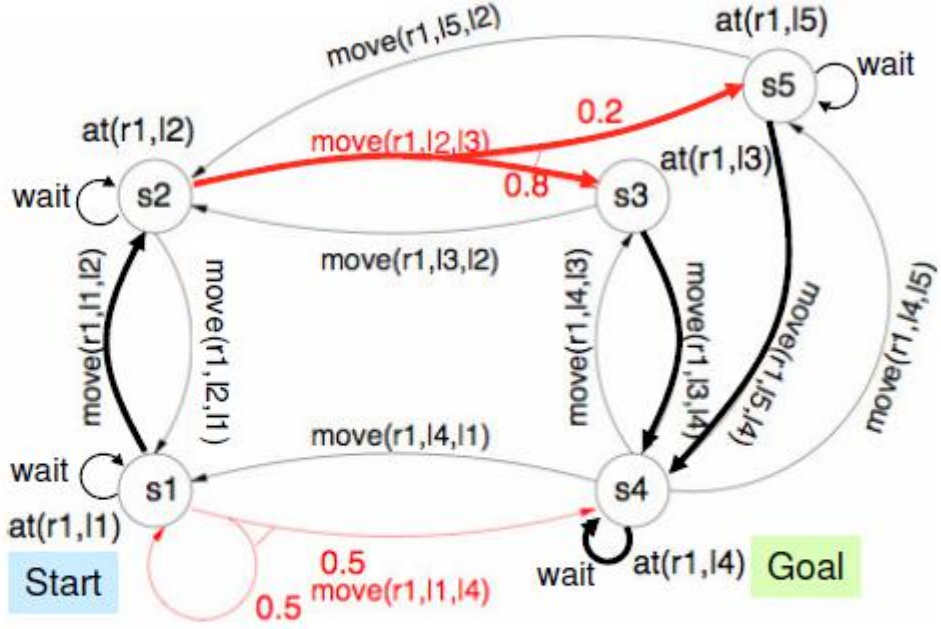
goal

h_1 = ⟨s1, s2, s3, s4, s4, … ⟩
h_2 = ⟨s1, s2, s5, s5 … ⟩

$$P(h_1 \mid \pi_1) = 1 \times 1 \times 0.8 \times 1 \times \ldots = 0.8$$
$$P(h_2 \mid \pi_1) = 1 \times 1 \times 0.2 \times 1 \times \ldots = 0.2$$
$$P(h \mid \pi_1) = 0 \text{ for all other } h$$

History: sequence of states – 3

$\pi_2 = \{$(s1, move(r1,l1,l2)),
(s2, move(r1,l2,l3)),
(s3, move(r1,l3,l4)),
(s4, wait),
(s5, move(r1,l5,l4))$\}$



goal

$h_1 = \langle$s1, s2, s3, s4, s4, … $\rangle$

$h_3 = \langle$s1, s2, s5, s4, s4, … $\rangle$

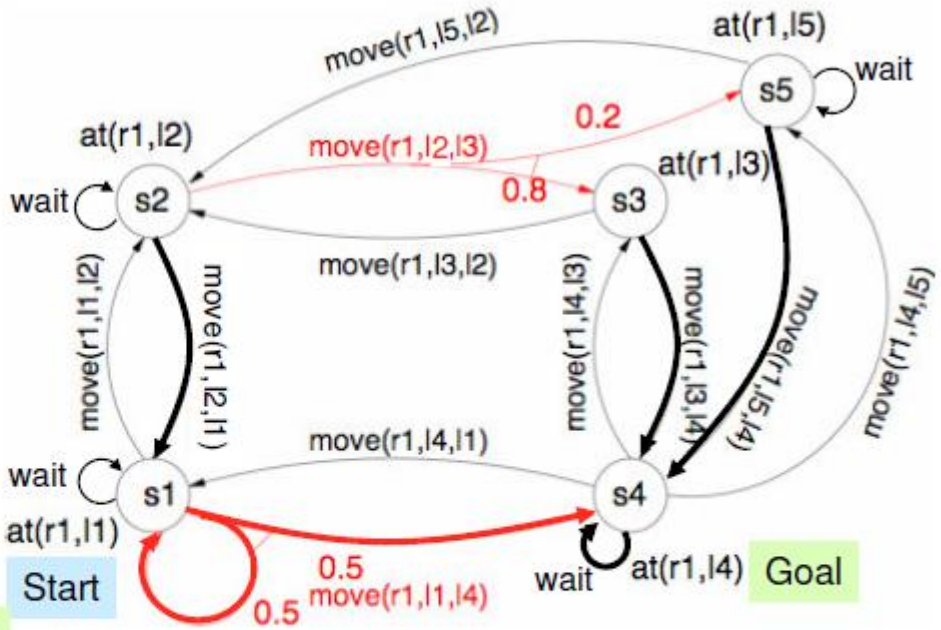
$P(h_1 \mid \pi_2) = 1 \times 0.8 \times 1 \times \ldots = 0.8$

$P(h_3 \mid \pi_2) = 1 \times 0.2 \times 1 \times \ldots = 0.2$

$P(h \mid \pi_2) = 0$ for all other h

History: sequence of states -4

π_3 = {(s1, move(r1,l1,l4)),
(s2, move(r1,l2,l1)),
(s3, move(r1,l3,l4)),
(s4, wait),
(s5, move(r1,l5,l4))}



goal

$h_4 = \langle s1, s4, s4, \ldots \rangle$ $P(h_4 \mid \pi_3) = 0.5 \times 1 \times 1 \times 1 \times 1 \times \ldots = 0.5$

$h_5 = \langle s1, s1, s4, s4, \ldots \rangle$ $P(h_5 \mid \pi_3) = 0.5 \times 0.5 \times 1 \times 1 \times 1 \times \ldots = 0.25$

$h_6 = \langle s1, s1, s1, s4, s4, \ldots \rangle$ $P(h_6 \mid \pi_3) = 0.5 \times 0.5 \times 0.5 \times 1 \times 1 \times \ldots = 0.125$

...

$h_7 = \langle s1, s1, s1, s1, s1, s1, \ldots \rangle$ $P(h_7 \mid \pi_3) = 0.5 \times 0.5 \times 0.5 \times 0.5 \times 0.5 \times \ldots = 0$

Qualidade de Políticas

- Em um PDM com transicões não deterministicas, Uma política pode garantir alcançar sempre o estado objetivo em um determinado número de passos ou custo ?

- Como definir quando uma política é melhor que outra? Chegar ao estado objetivo é o bastante?

- É necessário uma forma de medir a qualidade de uma dada política. Como ?

Qualidade de Políticas - 2

- Qual o valor de uma política? Valores são na verdade associados a históricos…

- Mas como vimos, políticas induzem uma distribuição de probabilidades sobre históricos. Assim…

- Essa qualidade (ou utilidade) pode ser medida através do valor esperado da adoção de uma polí

$$E[V_\pi(h)] = \sum_h P(h|\pi)\, V_\pi(h)$$

Política Ótima

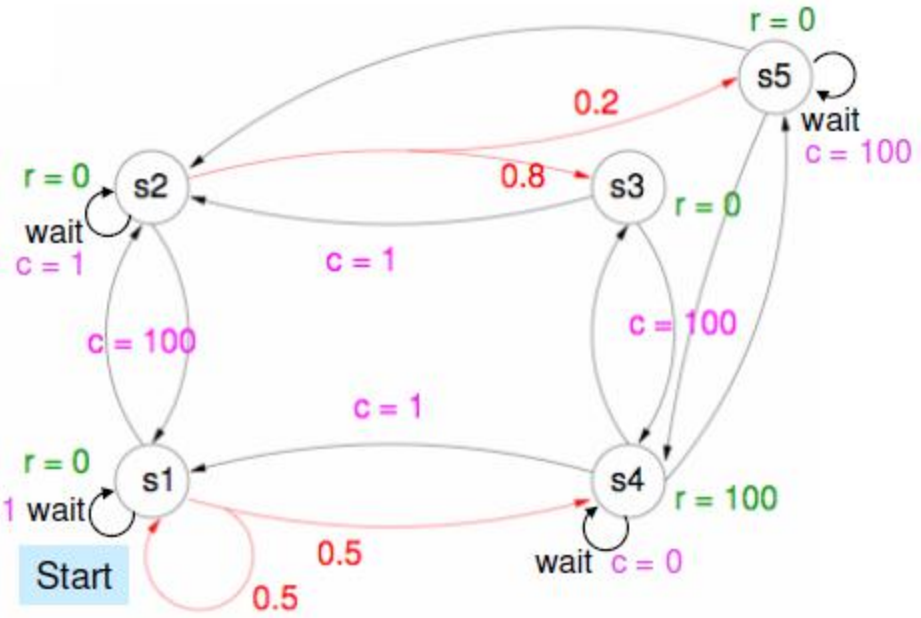
- Pode-se definir política ótima (π^*) como a política com o maior valor esperado.

$$E[V_\pi(h)] = \sum_h P(h|\pi)\, V_\pi(h)$$

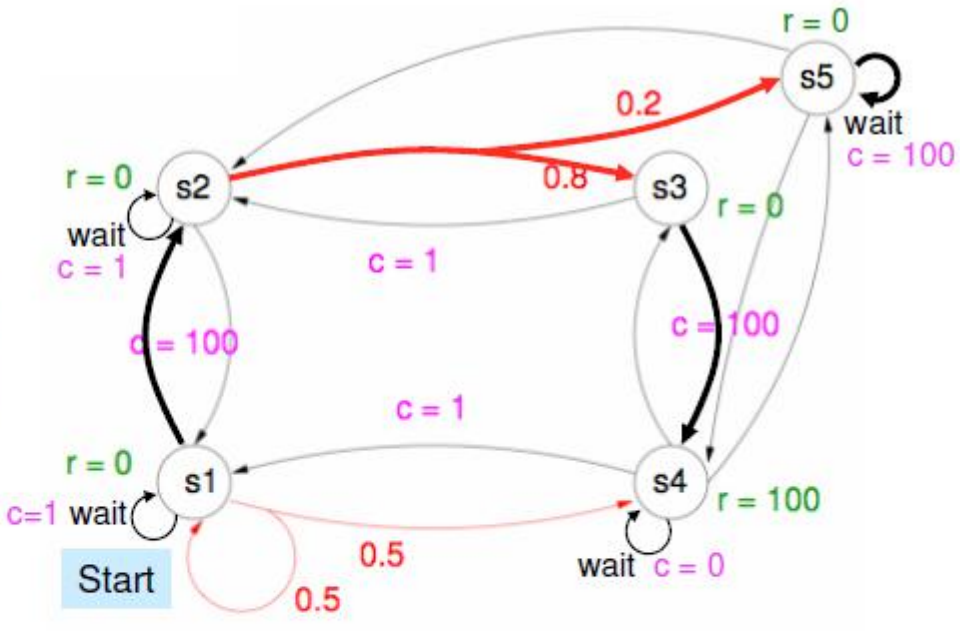
- Pergunta: Pode-se afirmar que ao adotar um **política ótima** um agente A sempre obterá maior valor que outro agente B com uma política **não ótima** em um dado período de tempo?

Reinforcements can be negative (cost) or positive (reward)

- Reinforcement $r(s)$ for each state s:
  - –: *cost* $C(s,a)$
  - +: *reward* $R(s)$
- Example:
  - $C(s,a) = 1$ for each "horizontal" action
  - $C(s,a) = 100$ for each "vertical" action
  - $C(s_1,wait) = 1$; $C(s_2,wait) = 1$; $C(s_4,wait) = 0$; $C(s_5,wait) = 100$
  - R as shown
- Utility function: generalization of a goal (additive rewards)
  - If $h = \langle s_0, s_1, \ldots \rangle$, then $V_\pi(h) = \sum_{i \geq 0} (R(s_i) - C(s_i, \pi(s_i)))$



$\pi_1 = \{(s1, move(r1,l1,l2)),$
$(s2, move(r1,l2,l3)),$
$(s3, move(r1,l3,l4)),$
$(s4, wait),$
$(s5, wait)\}$



$h_1 = \langle s1, s2, s3, s4, s4, \ldots \rangle$

$V_{\pi 1}(h_1) = (0 - 100) + (0 - 1) + (0 - 100) + 100 + 100 + \ldots = \infty$

$h_2 = \langle s1, s2, s5, s5 \ldots \rangle$

$V_{\pi 1}(h_2) = (0 - 100) + (0 - 1) + (0 - 100) + (0 - 100) + (0 - 100) + \ldots = -\infty$

- Algum problema com recompensa infinita?

Discounted Reinforcements

- Time should influence the value of a reinforcement?.
- A 100 dollars reward now or 100 dollars reward six months from now are the same?
- Problems with infinitive time require discounted reinforcements! Why?

- A **discount factor** γ describes the preference of an agent for current reinforcements over future reinforcements

$$0 \leq \gamma \leq 1$$

  - When γ is close to 0, reinforcements in the distant future are viewed as insignificant.
  - When γ is 1, discounted reinforcements are exactly equivalent to additive reinforcements.

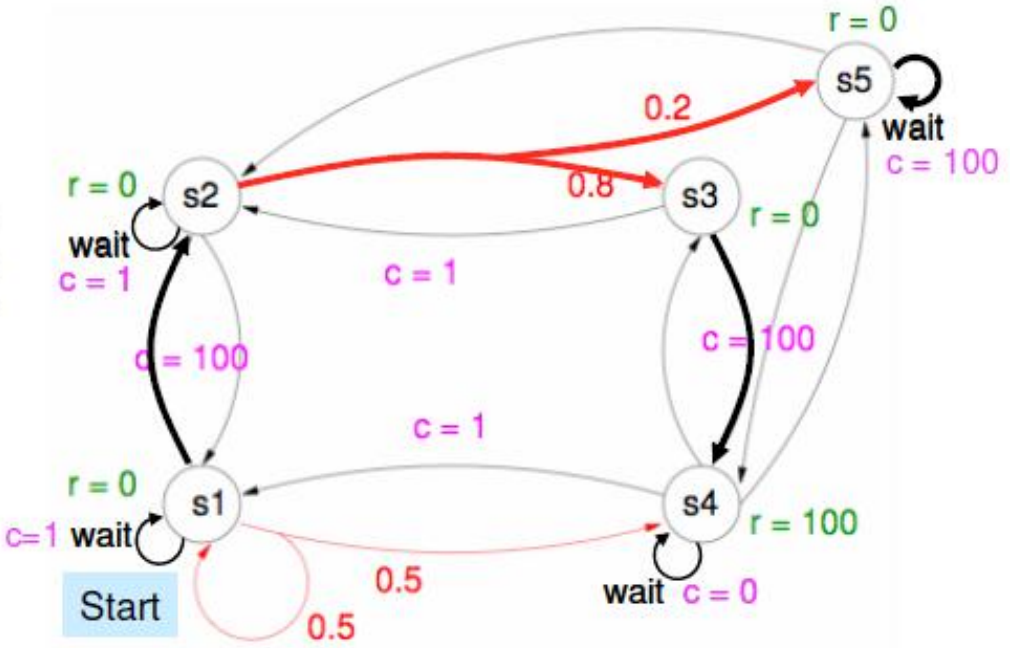
Discounted reinforcements:

$$V_\pi(h) = r(s_0) + \gamma r(s_1) + \gamma^2 r(s_2) + \gamma^3 r(s_3) + \ldots$$

Value with discounted reinforcement

$\pi_1 = \{(s1, move(r1,l1,l2)),$
$(s2, move(r1,l2,l3)),$
$(s3, move(r1,l3,l4)),$
$(s4, wait),$
$(s5, wait)\}$

$\gamma = 0.9$



$h_1 = \langle s1, s2, s3, s4, s4, \ldots \rangle$

$V_{\pi 1}(h_1) = .9^0(0-100) + .9^1(0-1) + .9^2(0-100) + .9^3\,100 + .9^4\,100 + \ldots = 547.9$

$h_2 = \langle s1, s2, s5, s5 \ldots \rangle$

$V_{\pi 1}(h_2) = .9^0(0-100) + .9^1(0-1) + .9^2(-100) + .9^3(-100) + \ldots = -910.1$

$E[V_{\pi 1}(h)] = 0.8 \times 547.9 + 0.2 \times (-910.1) = 256.3$

Otimalidade de Políticas e Horizonte

- A maximização do valor esperado é o critério mais utilizado para determinar otimização de políticas.
  - Entretanto, isso é dependente do número de passos (decisões) que o agente dispõe para agir. Isto é comumente chamado de horizonte de tomada de decisão.
  - O horizonte pode ser finito ou infinito

Finite Horizon: fixed time N after which nothing matters

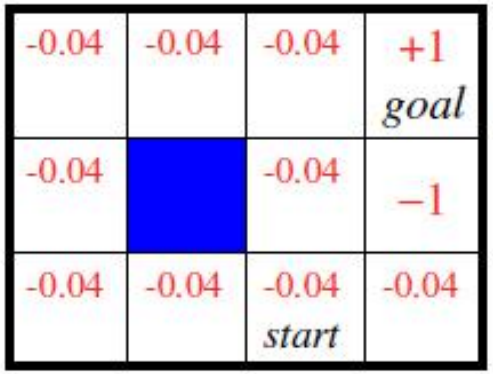
$$\forall M > N, \quad V_{\pi}(h)_{0:M} = V_{\pi}(h)_{0:N}$$

MDP de Horizonte Finito e Políticas Estacionárias

- Suppose an agent starts at (3,1) in the 4×3 environment, and N=3

  Then, to have any chance of reaching the goal, the optimal action is:

  $\pi(3,1)$=Up
- Suppose N=100. Then, there is plenty of time to take a safer route and go left: $\pi(3,1)$=Left



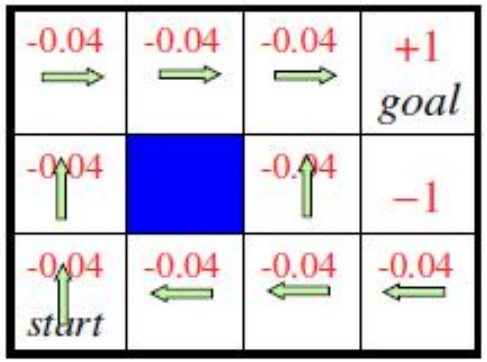
➔ The optimal action in a given state ***could change over time!***

Optimal policies for finite-horizon MDPs are nonstationary.

Stationary policies and Infinite Horizons

- With no fixed time limit, there is no reason to behave differently in the same state at different times → the optimal policy is **stationary**.

- For completely observed environments, policies for infinite horizon are simpler.
- Note that infinite horizon does not mean that all histories are infinite; it just means that there is no fixed deadline!
- In particular, there can be finite state sequences in an infinite-horizon MDP containing a terminal (absorbing) state.



Optimal policy for fully observable infinite-horizon MDP and r(s) as shown.

Policy Value with Infinite Horizon

1. Discounted reinforcements and $\gamma < 1$. If reinforcements are bounded by R_{max}:

$$V(h) = \sum_{t=0}^{\infty} \gamma^t r(s_t) \leq \sum_{t=0}^{\infty} \gamma^t R_{\max} = \frac{R_{\max}}{1-\gamma}$$

2. If the environment contains terminal states and if the agent is guaranteed to get one eventually.

   A policy that is guaranteed to reach a terminal state is a **proper policy** (in this case, we can use $\gamma = 1$).

3. You can compare infinite histories in terms of the average reinforcement obtained by time step.

Política Ótima

- Já vimos que uma política ótima (π^*) é aquele com maior valor esperado, então podemos definir:

$$\pi^* = \arg\max_{\pi} E\underbrace{\left[\sum_{t=0}^{\infty} \gamma^t r(s_t) \,\middle|\, \pi\right]}_{V_\pi(h)}$$

- Como encontrar uma politica ótima dado um MDP ?

An algorithm to calculate the optimal policy

- ***Basic idea: calculate the utility of each state and then use the state utilities to select an optimal action in each state.***

Utility of a state – defined in terms of the utility of state sequences:

$$V_\pi(s) = E\left[\sum_{t=0}^{\infty} \gamma^t r(s_t) \,\middle|\, \pi, s_0 = s\right]$$

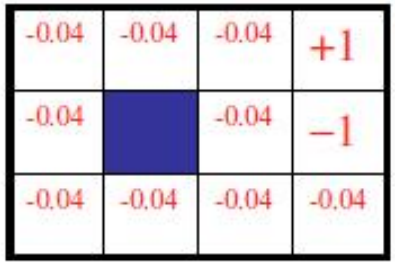
The true utility of a state, V(s), is just $V_{\pi^*}(s)$, which allows the agent to choose the action that maximizes the expected utility of the ***subsequent*** state:

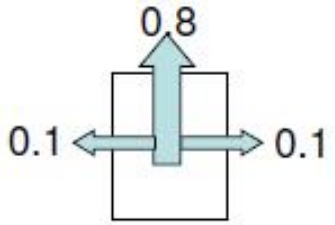
$$\pi^*(s) = \arg\max_a \sum_{s'} p(s' \mid s, a) V(s')$$

Value Iteration Algorithm

- *The utility of a state is the immediate reinforcement for that state plus the expected discounted utility of the next state, assuming that the agent chooses the optimal action:*

$$V(s) = r(s) + \gamma \max_a \sum_{s'} p(s' \mid s, a) V(s') \quad \textbf{Bellman equation}$$





$V(1,1) = -0.04 + \gamma \max\{$
$0.8V(1,2)+0.1V(2,1)+0.1V(1,1),$ (Up)
$0.9V(1,1)+0.1V(1,2),$ (Left)
$0.9V(1,1)+0.1V(2,1),$ (Down)
$0.8V(2,1)+0.1V(1,2)+0.1V(1,1)\}$ (Right)

$V(3,3) = -0.04 + \gamma \max\{$
$0.8V(3,3)+0.1V(2,3)+0.1V(4,3),$ (Up)
$0.8V(2,3)+0.1V(3,2)+0.1V(3,3),$ (Left)
$0.8V(3,2)+0.1V(2,3)+0.1V(4,3),$ (Down)
$0.8V(4,3)+0.1V(3,3)+0.1V(3,2)\}$ (Right)

Value Iteration Algorithm - 2

- n possible states ➔ n Bellman equations, one for each state
  - » We want to solve it simultaneously!
    - Problem: equations are *nonlinear*!
    - Solution: iterative approach!

◆ **Value iteration algorithm**:

- » Start with arbitrary initial values for the utilities
- » Calculate the right-hand side of the equation and plug it into the left-hand side
- » Repeat until reach equilibrium

$$V_{i+1}(s) \leftarrow r(s) + \gamma \max_a \sum_{s'} p(s' \mid s, a) V_i(s') \quad \textbf{Bellman update}$$

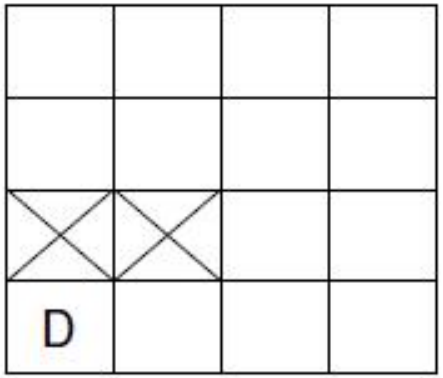
or

$$V_{i+1}(s) \leftarrow \max_a r(s, a) + \sum_{s'} \gamma\, p(s' \mid s, a) V_i(s')$$

Exemplo Deterministico – Função de Valor

- Ambiente discreto 4x4, com obstáculos.
- Agente deve alcançar posição destino D a partir de qualquer lugar do ambiente.
- Ações que o agente pode realizar: N, S, L, O
- Penalidade por executar uma ação (qualquer) = -1
  - Melhor política => caminho mais curto
- Exemplo para $\gamma = 1$ e $p(s' \mid s,a) = 1 \ \forall s', s, a$, sendo s' vizinho de s (vizinhança 4 na grade); $p(s' \mid s,a) = 0$ para estados não vizinhos

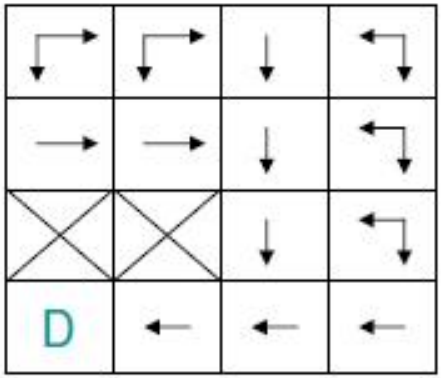
Exemplo - 2



Ambiente exemplo

-7	-6	-5	-6
-6	-5	-4	-5
		-3	-4
D 0	-1	-2	-3

Função valor ótima: indica as penalidades esperadas até atingir o destino, seguindo uma política ótima.

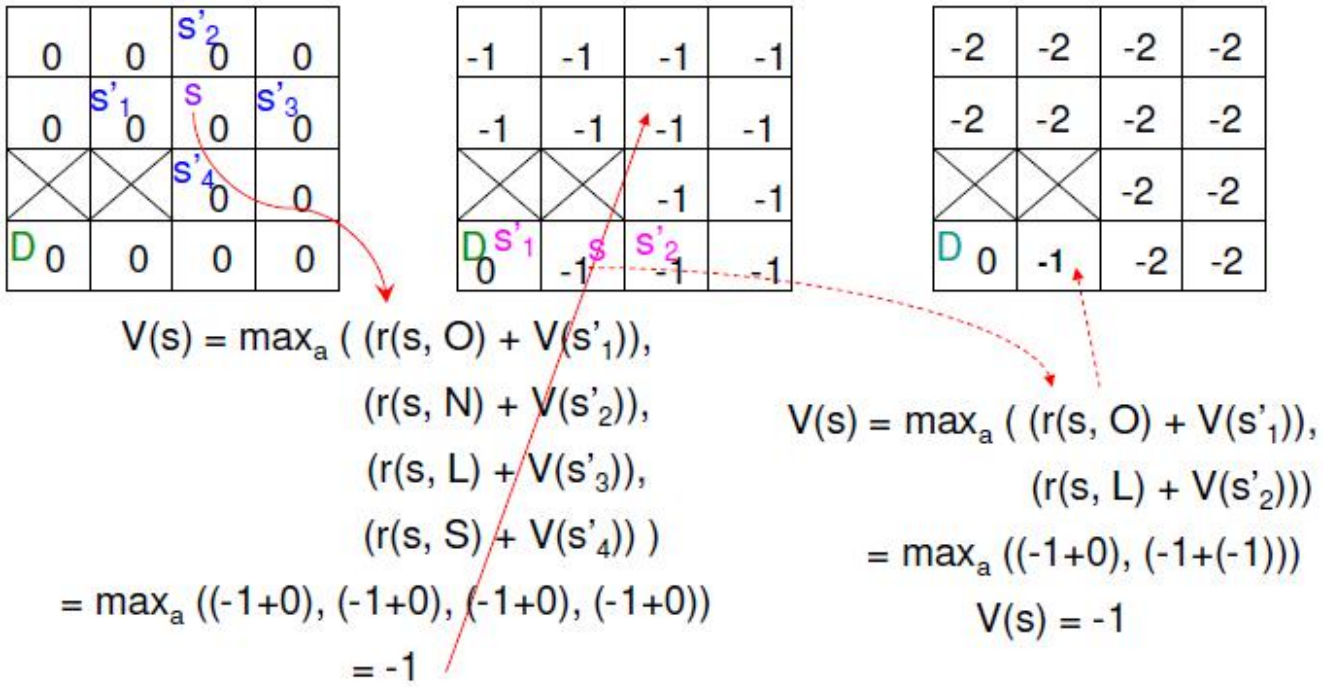


Política ótima obtida a partir da função valor ótima (melhores ações, para cada estado).

Calculando V(s)

- Considerando r(s,a) e fator de desconto igual a 1

$$V_{i+1}(s) \leftarrow \max_a r(s,a) + \sum_{s'} \gamma\, p(s'|s,a) V_i(s')$$



Outro Exemplo Gridworld

Agente move-se no grid.

Observações = estados = células.

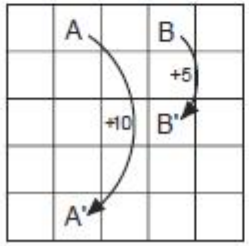
Quatro ações possíveis: N, S, L, O.

Ações que produzem choque com paredes: $r = -1$, agente não se move.

Estado A: reforço $+10$, qualquer ação leva a A'.

Estado B: reforço $+5$, qualquer ação leva a B'.

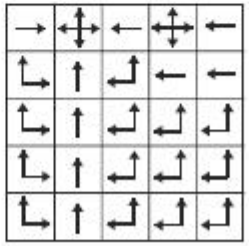
Política de ações: para cada estado, uma ação entre as quatro é escolhida aleatoriamente.



a) gridworld

22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7

b) V^*



c) π^*

Discussão da Iteração de Valor

- Algoritmo de Iteração de valor computa um novo valor a cada iteração e escolhe a política baseado nesses valores
  - Este algoritmo converge em número de iterações em tempo polinomial do número de estados
- O número de estados pode ser muito grande em problemas reais e É necessário examinar o espaço inteiro em cada iteração. Por tal razão, o algoritmo demanda significativa quantidade de tempo e espaço para problemas com grande número de de estados
- Há algoritmos alternativos como iteração de política
- Além disso, a **função de transição propabilística p** e os retornos devem ser conhecidos, mas muitas vezes não é este o caso!

A equação de Bellman e o algoritmo de Iteração de Valor

- Vimos que uma política ótima é aquela que maximiza o valor esperado de seguir de tal política a partir de um estado inicial. Formalmente:

$$\pi^* = \arg\max_{\pi} E\underbrace{\left[\sum_{t=0}^{\infty} \gamma^t r(s_t) \,\middle|\, \pi\right]}_{V_\pi(h)}$$

- Vimos também que

$$\pi^*(s) = \arg\max_{a} \sum_{s'} p(s' \mid s, a) V(s')$$

Programação Dinâmica e Equações de Otimalidade de Bellman

- Considere os seguintes operadores de Programação Dinâmica

O Operador de Aproximações Sucessivas T_μ. Para qualquer função $V : \mathcal{S} \mapsto \mathbb{R}$ e política $\mu : \mathcal{S} \mapsto \mathcal{A}$,

$$(T_\mu V)(s) = r(s, \mu(s)) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s, \mu(s)) V(s')$$

O Operador de Iteração de Valores T. Para qualquer função $V : \mathcal{S} \mapsto \mathbb{R}$,

$$(TV)(s) = \max_a [r(s, a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s, a) V(s')]$$

Teoremas de Programação Dinâmica

Para qualquer $V(s)$ e estado s, a função de utilidade ótima satisfaz

$$V^*(s) = \lim_{N\to\infty}(T^N V)(s)$$

Para toda política μ e estado s, a função de utilidade satisfaz

$$V_\mu(s) = \lim_{N\to\infty}(T_\mu^N V)(s)$$

- É possível demonstrar que a função V* é o ponto fixo do operador T (Equação de otimalidade de Bellman), ou

$$V^* = TV^*$$

- Também é possível demonstrar que para uma dada política μ, a função V_μ é a única função limitada que satisfaz

$$V_\mu = T_\mu V_\mu$$

Algoritmos para MDP

- O algoritmo de Iteração de valor, nada mais é do que a aplicação recursiva do operador T, a uma aproximação inicial arbitrária V_0

$$\mu^*(s) = \arg[\lim_{N\to\infty}(T^N V)(s)]$$

ou seja,

$$\mu^*(s) = \arg\max_a[r(s,a) + \gamma \sum_{s'\in S} P(s'|s,a)V^*(s')]$$

Outro algoritmo para MDP: Iteração de Política

Algoritmo 2: Método da Iteração de Políticas

Dada uma política inicial μ^0, dois passos sucessivos são executados em *loop* até que μ^* (ou uma boa aproximação) seja encontrada.

No primeiro passo, a política atual μ_k é avaliada ($i.e.$, tem sua utilidade calculada), através da solução do sistema:

$$V_{\mu_k} = T_{\mu_k} V_{\mu_k}$$

Este é o passo de *Avaliação da Política.*

No segundo passo, uma política melhorada é obtida através de:

$$\mu_{k+1}(s) = \arg[(TV_{\mu_k})(s)]$$

ou seja

$$\mu_{k+1}(s) = \arg\min_a [s, a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s, a) V_{\mu_k}(s')]$$

Outro Exemplo Gridworld

Agente move-se no grid.

Observações = estados = células.

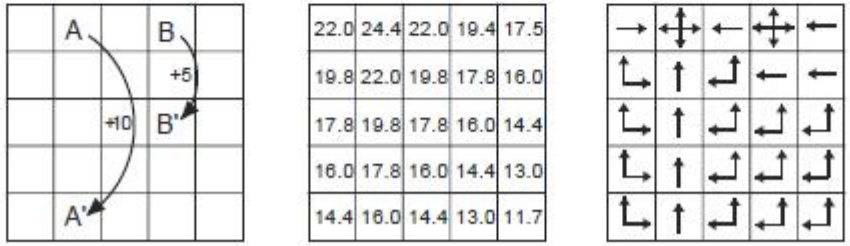
Quatro ações possíveis: N, S, L, O.

Ações que produzem choque com paredes: $r = -1$, agente não se move.

Estado A: reforço $+10$, qualquer ação leva a A'.

Estado B: reforço $+5$, qualquer ação leva a B'.

Política de ações: para cada estado, uma ação entre as quatro é escolhida aleatoriamente.



a) gridworld b) V^* c) π^*

- Mais informações Cap. 21 (Russel e Norvig, 2013)