PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

INTELIGENCIA ARTIFICIAL (INF371)

APRENDIZAJE POR REFUERZO (PARTE 1)

Dr. Edwin Villanueva Talavera

Contenido

- Procesos de decisiones de Markov (PDM)
- Iteración de Valor
- Iteración de Política

Bibliografía:

Capitulo 17.1, 17.2, 17.3, 21.1, 21.2 del libro:

Stuart Russell & Peter Norvig "Artificial Intelligence: A modern Approach", Prentice Hall, Third Edition, 2010

Procesos de decisiones de Markov

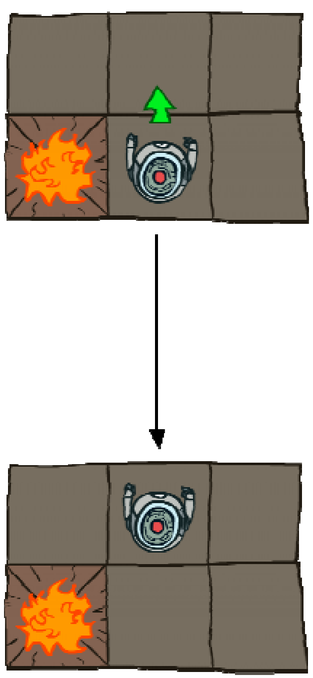
Características:

- Son un tipo de problema de toma de decisiones secuenciales
- La utilidad del agente depende de una secuencia de decisiones.
- El ambiente es No determinístico: el resultado de la acción del agente es incierta
- Modelo de transición probabilístico: El resultado de cada acción resulta en otro estado con cierta probabilidad
- Transiciones Markovianas: probabilidades de alcanzar un nuevo estado solo dependen del estado actual, no del pasado.
- Existe una función de recompensa en cada estado s, R(s)

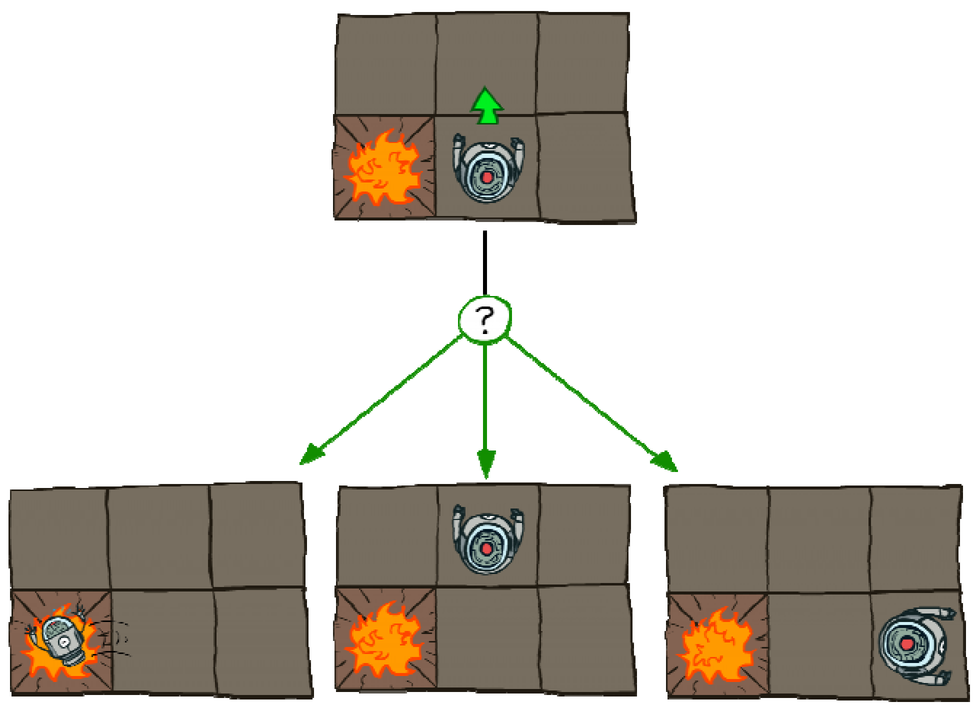
Procesos de decisiones de Markov

Diferencia entre ambiente determinístico y un PDM:

Deterministic Grid World



Stochastic Grid World



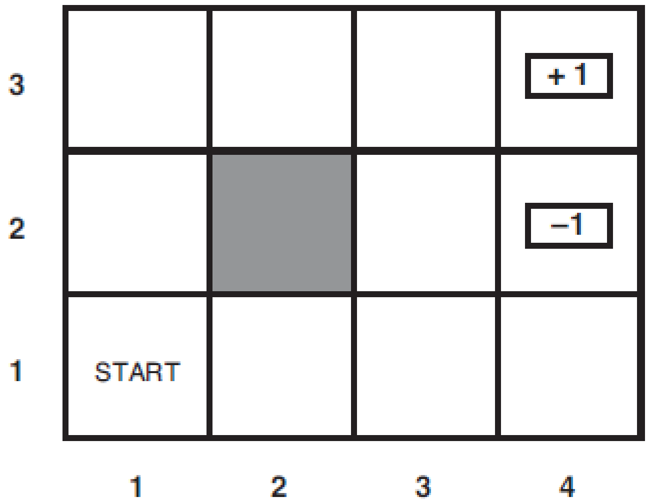
Procesos de decisiones de Markov

Definición de un PDM:

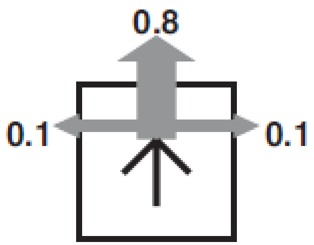
- un estado inicial s_o
- un conjunto de estados $s \in S$
- un conjunto de acciones en cada estado: **Actions(s)**
- un modelo de transición de estados: P(s' | s, a)
  - probabilidad de alcanzarse *s*' desde *s* ejecutando acción *a*.
  - Propiedad de Markov: esa probabilidad depende apenas de *s* y *a* y no del histórico de estados y acciones
- una función de recompensa $R(s)$
- (Tal vez) uno o mas estados terminales

Procesos de decisiones de Markov

Ejemplo:



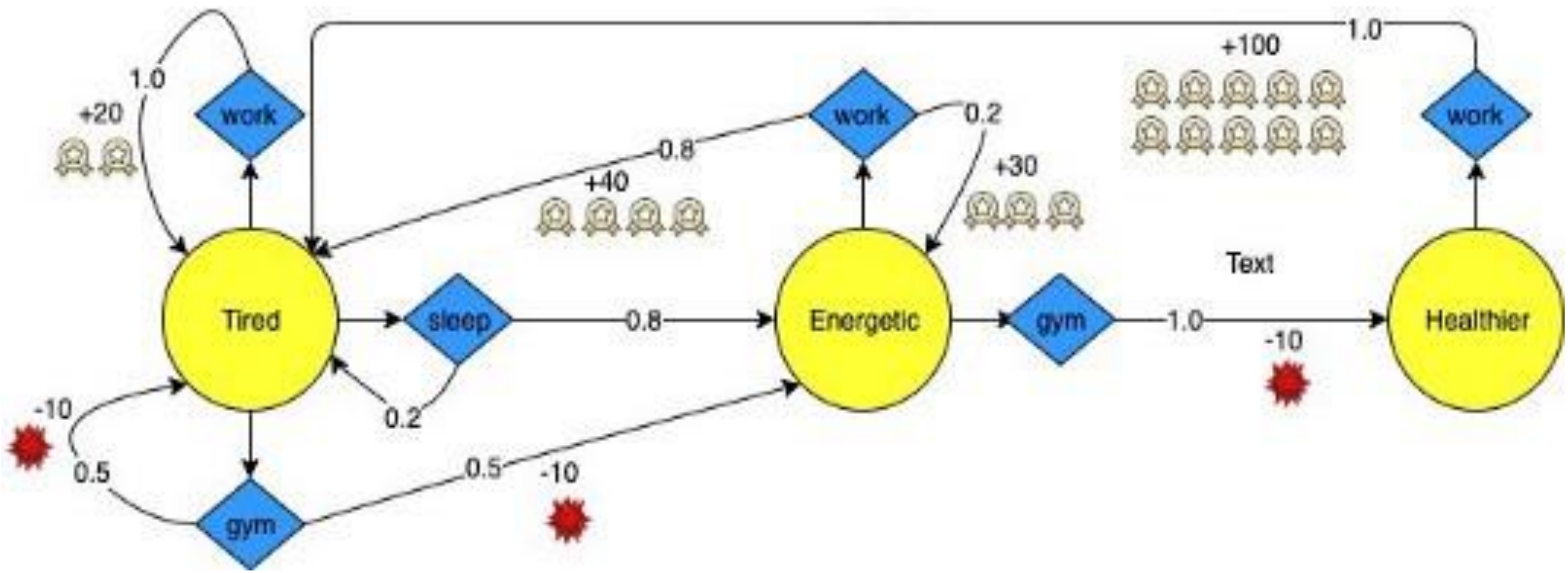
- Estado inicial: START
- Acciones: Ir Arriba, Abajo, Izquierda, Derecha
- Modelo de transición: P(s' | s, a)
  - El agente tiene probabilidad de 0.8 de moverse en la dirección pretendida y 0.2 de moverse a los estados laterales.
- Función de recompensa: R(s)
  - Estados terminales tienen recompensa: +1 y -1
  - Todos los otros estados tienen recompensa: -.04



Si no hubiese incerteza, podríamos usar búsqueda para encontrar la solución óptima.

Procesos de decisiones de Markov

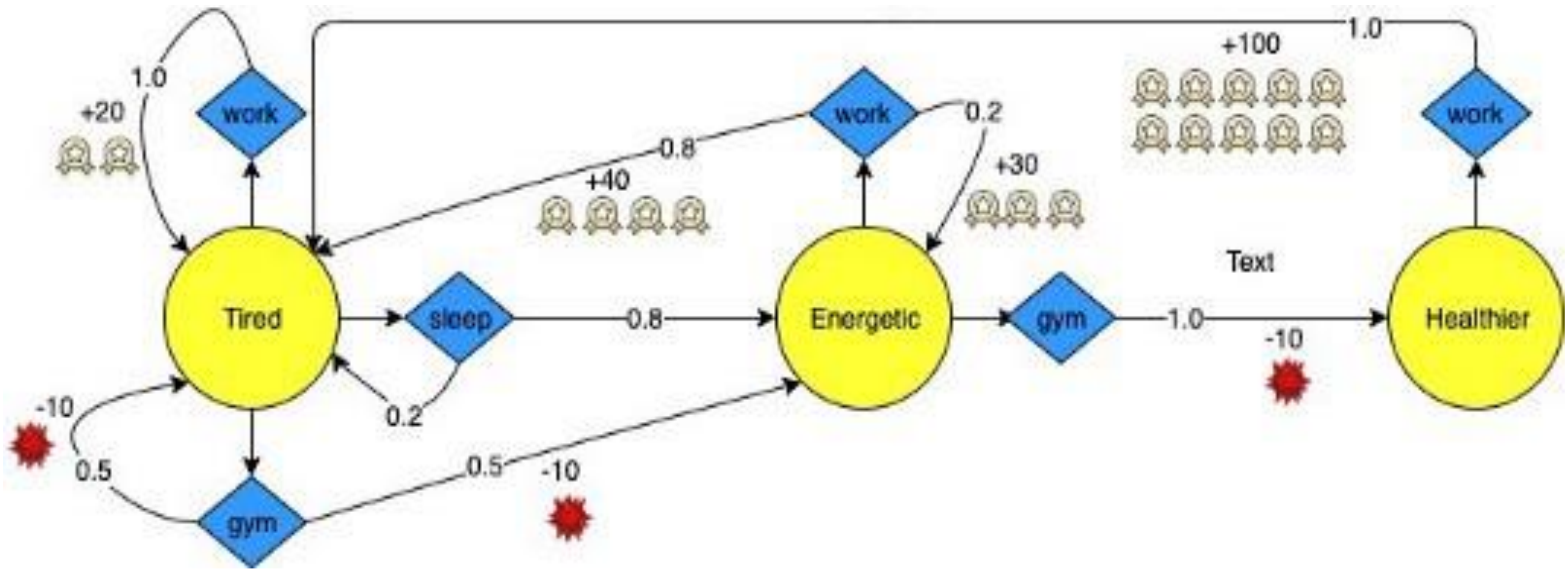
Ejemplo: https://medium.com/ai%C2%B3-theory-practice-business/reinforcement-learning-part-3-the-markov-decision-process-9f5066e073a2



- Estados: Tired, Energetic, Healthier
- Acciones: work, sleep, gym
- Modelo de transición: números en las aristas (probabilidades)
- Función de recompensa: Dinero ganado/perdido en cada estado-acccion

Procesos de decisiones de Markov

Cuál es la secuencia de acciones optima para ganar dinero?



política

Procesos de decisiones de Markov

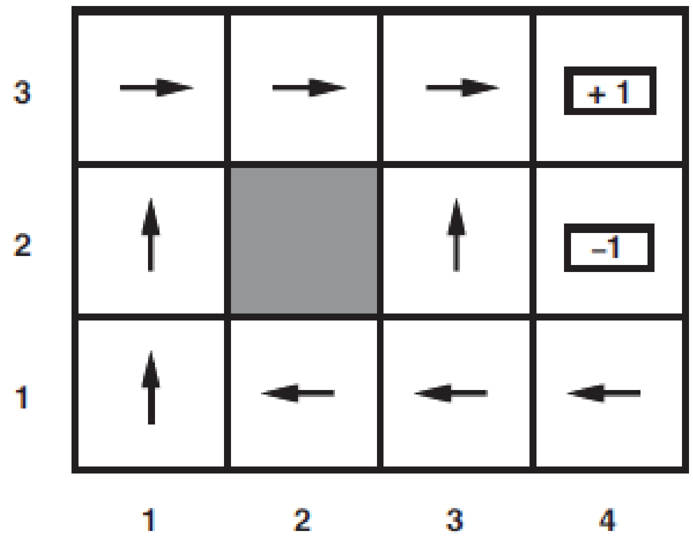
Soluciones de PDMs:

- En un ambiente determinístico con un único agente, la solución es un **plan** = secuencia de acciones óptima.
- En un PDM, la solución es una **política** (denotada por $\pi(s)$) = especifica una acción para cada estado.
  - La **política óptima** es la que produce una Utilidad esperada mas alta posible.
- Si el agente tuviese una política completa, independiente del resultado de sus acciones, el sabría que hacer enseguida.
- Cada vez que una política es ejecutada a partir del estado inicial, la naturaleza estocástica del ambiente llevará a un histórico diferente.

Procesos de decisiones de Markov

Ejemplo:

- Política óptima para recompensa en estados no-terminales $R(s)$ = -.04



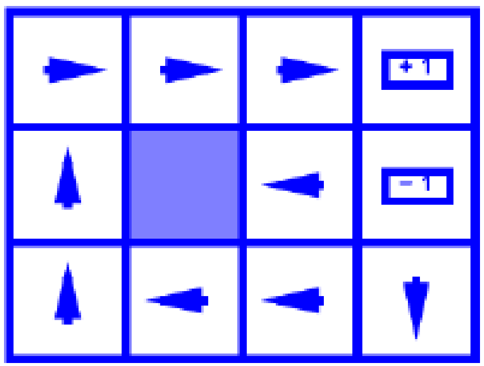
Procesos de decisiones de Markov

Equilibrio Riesgo-Recompensa:

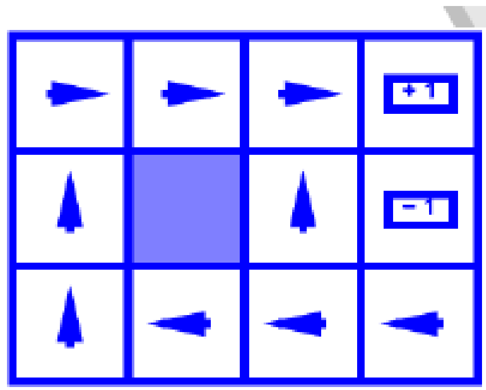
- El equilibrio entre riesgo y recompensa cambia dependiendo del valor R(s) para los estados no terminales.
- El mantenimiento de un equilibrio cuidadoso entre riesgo y recompensa es una característica de los PDMs que no surge en problemas de búsqueda determinística.

Ejemplo:

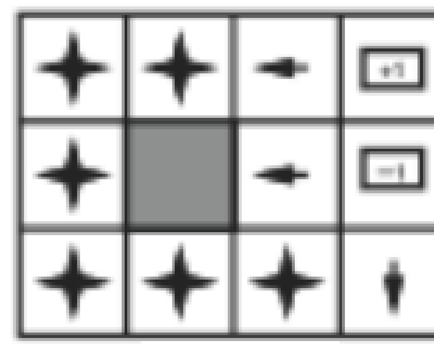
- Políticas óptima para diferentes recompensas de estados no terminales



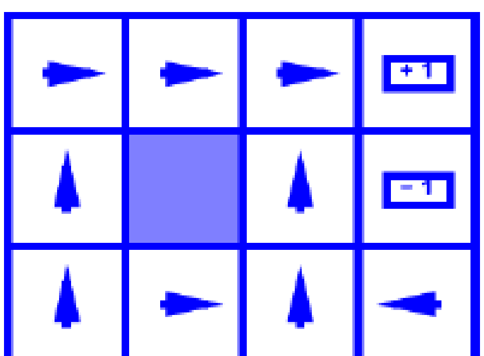
R(s) = -0.01



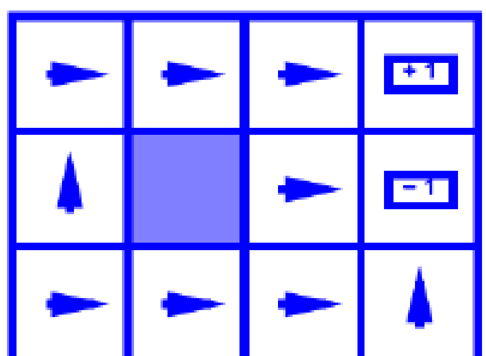
R(s) = -0.03



R(s)>0



R(s) = -0.4



R(s) = -2.0

Procesos de decisiones de Markov

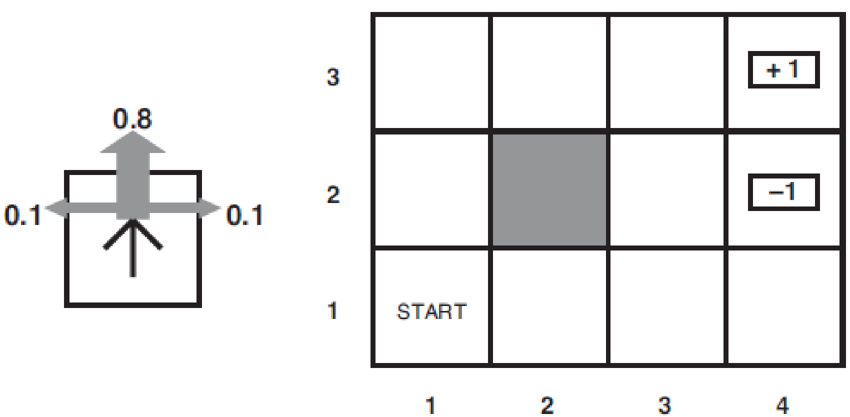
Utilidades de las secuencias:

- Para medir el desempeño de un agente que visita una secuencia de estados $[s_0, s_1, \ldots, s_n]$ en un PDM se usa una función de utilidad
- Comúnmente se usa la notación $U_h([s_0, s_1, \ldots, s_n])$
- En el ejemplo, la función de utilidad era la suma de las recompensa de cada estado, pero esa no es la única posibilidad.

Procesos de decisiones de Markov

Consideraciones en Utilidades: Horizonte

- Para definir utilidades en PDM hay que definir que tipo de horizonte se tendrá en la toma de decisiones:
  - Horizonte finito: existe un tiempo fijo N después del cual nada importa: GAME OVER, o también $U_h([s_0, s_1, \ldots, s_{N+k}]) = U_h([s_0, s_1, \ldots, s_N])$, k>0
  - Horizonte infinito: No hay tiempo fijo par seguir una ruta segura



Ejemplo: Inicio en (3,1) y N=3

- Para tener oportunidades de alcanzar el estado +1, el agente debe ir directamente a él (ejm. ir para arriba aun arriesgando caer en -1)

Con horizonte infinito, la acción optima depende apenas del estado actual: **política óptima es estacionaria.** Se hará esta suposición en lo sucesivo

Consideraciones en Utilidades: Preferencias Estacionarias

- Se dice que el agente tiene preferencias estacionarias:
  - Si al preferir una de las secuencias: $[s_0, s_1, s_2, \ldots]$ o $[s_0, s'_1, s'_2, \ldots]$ en estado s_0, en un estado posterior preferirá la misma secuencia de estados: $[s_1, s_2, \ldots]$ o $[s'_1, s'_2, \ldots]$
  - Si debe escoger entre un futuro u otro que comience mañana, entonces debe preferir ese mismo futuro aun así comience hoy!

Procesos de decisiones de Markov

Utilidades con Horizontes Infinitos y Preferencias Estacionarias

- Hay dos posibilidades para definir utilidades cuando el agente tiene horizontes infinitos y preferencias estacionarias:
  - Utilidad con Recompensas Aditivas:

$$U([s_0, \dots, s_n]) = \mathrm{R}(s_0) + R(s_1) + \cdots$$

  - Utilidad con Recompensas Descontadas:

$$U([s_0, \dots, s_n]) = \mathrm{R}(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots$$

donde γ es un valor entre 0 y 1, llamado **factor de descuento,** que describe la preferencia de un agente por recompensas actuales en lugar de recompensas futuras.

Procesos de decisiones de Markov

Utilidades con Horizontes Infinitos y Preferencias Estacionarias

- Con utilidades de recompensa aditivas se puede tener utilidades infinitas cuando no existe estados terminales, o el agente no encuentra uno de ellos.
- Utilidades infinitas no permiten encontrar políticas optimas. Existen las siguientes soluciones:
  - Garantizar que el agente siempre alcance un estado terminal (política propia)
  - Usar recompensas descontadas: (Esta es la mejor solución)

$$U_h([s_0, s_1, s_2, \ldots]) = \sum_{t=0}^{\infty} \gamma^t R(s_t)$$

Cuanto menor el valor de γ menor el "horizonte" (mas se penaliza recompensas futuras)

Procesos de decisiones de Markov

Evaluación de Políticas

- Dada la naturaleza no determinista de PDM, una política no genera una única secuencia de estados, si no un conjunto de secuencias que difieren según el modelo de transición
- Para evaluar una política π podemos usar el valor esperado de la utilidad de las secuencias (recompensas descontadas) que puede generar la política. En el estado **s** este valor seria:

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right]$$

donde S_t es una variable aleatoria que representa el estado del agente en el tiempo t cuando se ejecuta la política π empezando en ***s***

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

Política Optima

- La acción recomendada en s por una política óptima, $\pi^*(s)$, es aquella que genera el mas alto valor $U^\pi(s)$ de todas las políticas posibles que comienzan en s:

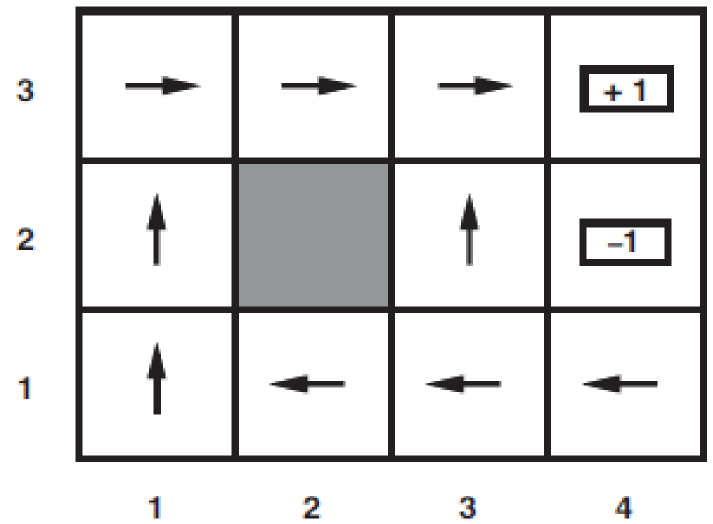
$$\pi_s^* = \operatorname*{argmax}_\pi U^\pi(s)$$

- Definimos la utilidad de un estado *s* como el valor $U(s) = U^{\pi^*}(s)$ esto es, la expectativa de la suma de recompensas descontadas a partir de s dado que el agente ejecuta una política optima

Ejemplo de Utilidades

3	0.812	0.868	0.918	+ 1
2	0.762		0.660	–1
1	0.705	0.655	0.611	0.388
	1	2	3	4

Utilidades (com politica óptima)



Politica óptima

No es lo mismo Utilidad de um estado (U(s)) que Recompensa del estado (R(s) !

En la práctica, Cómo encontramos la política optima?

Recuerde, la política debe especificar una acción para cada estado

Procesos de decisiones de Markov

Ecuación de Bellman:

- Ecuación recursiva definiendo la utilidad de un estado:

$$U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s')$$

es la recompensa inmediata correspondiente a ese estado + la utilidad descontada esperada del próximo estado, suponiendo que el agente escoja la acción óptima

- Al resolver la ecuación de Bellman y obtener las utilidades *U(s)* para cada estado **s** se puede encontrar la política optima:

$$\pi^*(s) = \operatorname*{argmax}_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s')$$

Ejercicio Ecuación de Bellman:

3	0.812	0.868	0.918	$\boxed{+1}$
2	0.762		0.660	$\boxed{-1}$
1	0.705	0.655	0.611	0.388
	1	2	3	4

Utilidades encontradas al resolver la ecuación de Bellman con R(s) = 0.04, $\gamma=1$

¿Que acción ejecutar en (1,1)?

$$
\begin{aligned}
U(1,1) = -0.04 + \gamma \max\{\ & 0.8U(1,2) + 0.1U(2,1) + 0.1U(1,1), && \text{Arriba} \\
& 0.9U(1,1) + 0.1U(1,2), && \text{Izquierda} \\
& 0.9U(1,1) + 0.1U(2,1), && \text{Abajo} \\
& 0.8U(2,1) + 0.1U(1,2) + 0.1U(1,1)\ \} && \text{Derecha}
\end{aligned}
$$

Arriba seria la acción recomendada por la política optima.

Procesos de decisiones de Markov

Resolviendo la Ecuación de Bellman:

- ¿Por que no usar algoritmos de búsqueda?
  - Árbol puede ser infinito
  - Tendríamos que hacer una búsqueda para cada estado
  - Repite muchas veces los mismos cálculos siempre que el mismo estado fuese alcanzado.
- Idea: **Iteración de valor**
  - Calcular valores de utilidad óptimos para todos los estados simultáneamente, usando aproximaciones sucesivas.

Iteración de Valor

Algoritmo de Iteración de Valor:

- Si existe n estados posibles, tendremos n ecuaciones de Bellman (una para cada estado) con n incógnitas (las funciones de utilidad).
- Para encontrar la política optima tenemos que resolver ese sistema de ecuaciones NO Lineares (por el operador max).

Iteración de Valor

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

Algoritmo de Iteración de Valor:

- Valores iniciales arbitrarios para las utilidades $U_0(s)$
- Repetir hasta llegar al equilibrio
  - Calcular el lado derecho de la ecuación de Bellman
  - Actualizar la utilidad de cada estado a partir de la utilidad de sus vecinos

Lo que se hace en cada iteración es la actualización de Bellman:

$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U_i(s')$$

- La convergencia es garantizada con un horizonte finito o recompensas descontadas
- Los valores finales serán soluciones para las ecuaciones de Bellman

Iteración de Valor

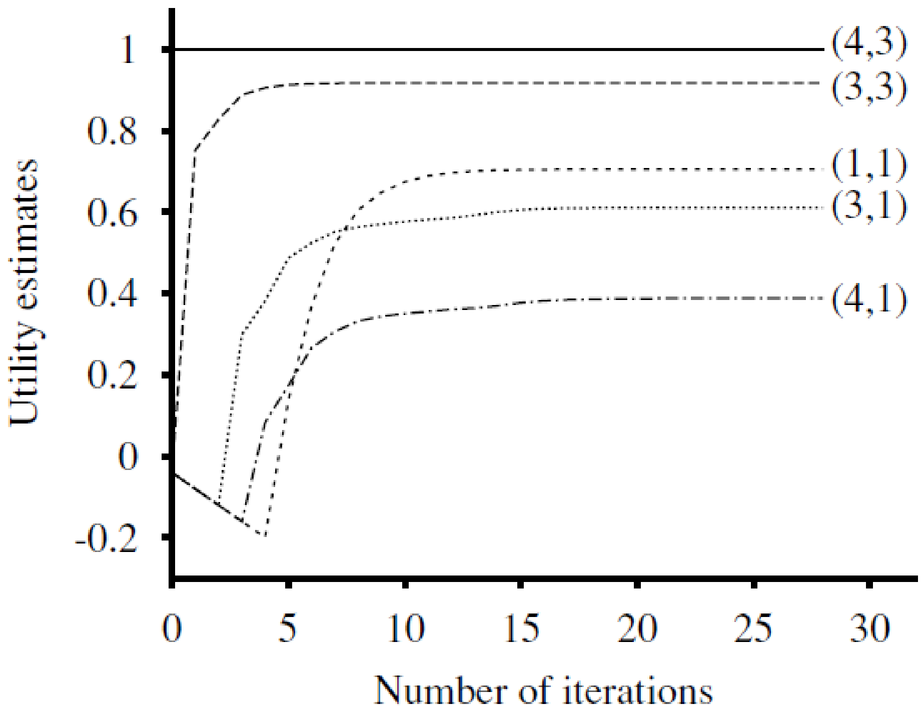
Algoritmo de Iteración de Valor:

function VALUE-ITERATION(mdp, ϵ) **returns** a utility function
  inputs: mdp, an MDP with states S, actions $A(s)$, transition model $P(s' \mid s, a)$,
    rewards $R(s)$, discount γ
    ϵ, the maximum error allowed in the utility of any state
  local variables: U, U', vectors of utilities for states in S, initially zero
    δ, the maximum change in the utility of any state in an iteration

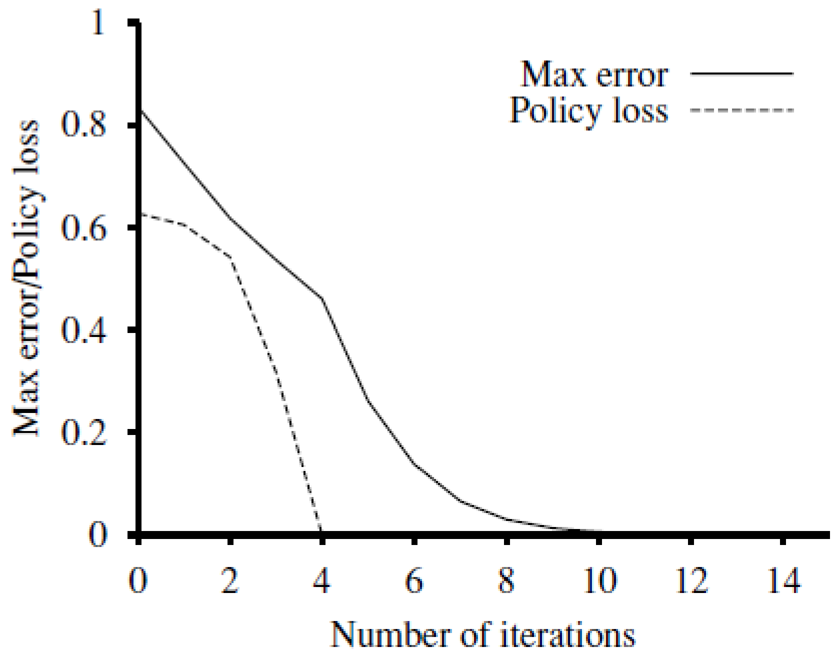
  repeat
    $U \leftarrow U'$; $\delta \leftarrow 0$
    for each state s **in** S **do**
      $U'[s] \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\, U[s']$
      if $|U'[s] - U[s]| > \delta$ **then** $\delta \leftarrow |U'[s] - U[s]|$
  until $\delta < \epsilon(1-\gamma)/\gamma$
  return U

Iteración de Valor

Evolución de utilidades y politicas en el ejemplo de PDM con el algoritmo de Iteración de Valor:



Evolución de Utilidades

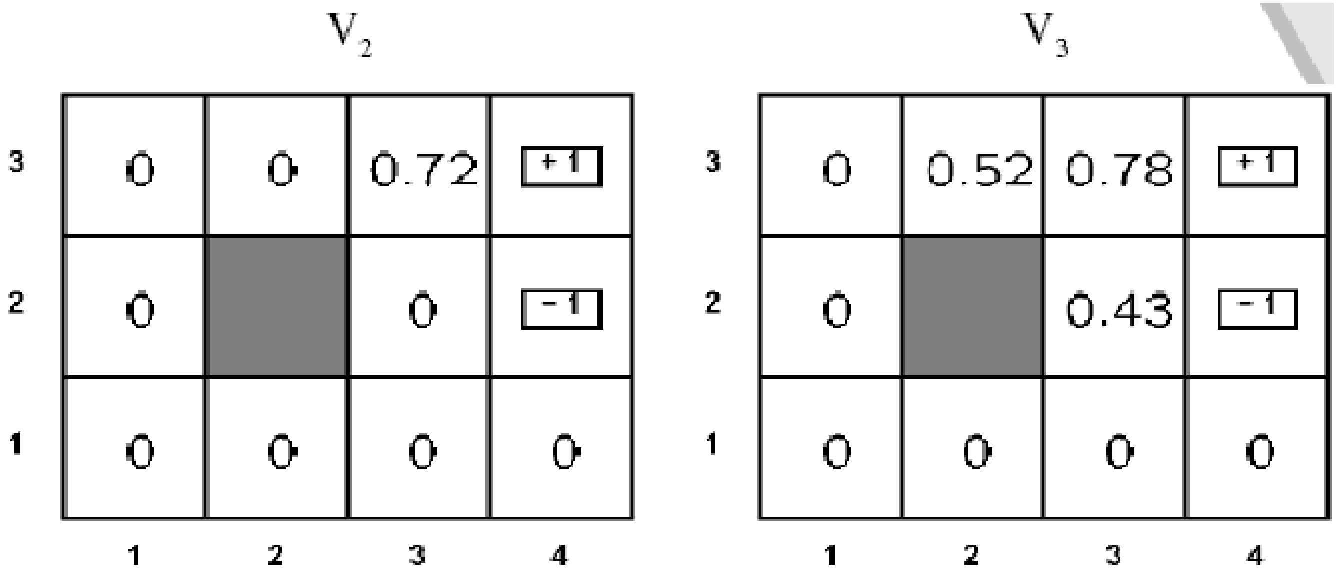


Evolución de maximo Error: $||U_i - U||$

Evolución de Error de Politica: $||U^{\pi_i} - U||$

Iteración de Valor

Evolución de utilidades en el ejemplo de PDM con el algoritmo de Iteración de Valor:



La información se propaga para fuera a partir de los estados terminales.

Iteración de Política

Fundamentos:

- Muchas veces no se necesita una estimación precisa de las utilidades para generar una política óptima
- Iteración de Política se basa en dicha idea para simplificar el cálculo de función de utilidades
- Empezando con una política inicial π_0 se itera:
  - Evaluación de política: dada la política π_i se calcula la utilidad U_i (sin usar el operador MAX) para cada estado
  - Mejoramiento de política: Calcular la nueva política π_{i+1} maximizando la utilidad esperada (calculada en base a U_i)

$$\pi_{i+1}(s) = \underset{a \in A(s)}{\operatorname{argmax}} \sum_{s'} P(s' \mid s, a) U_i(s')$$

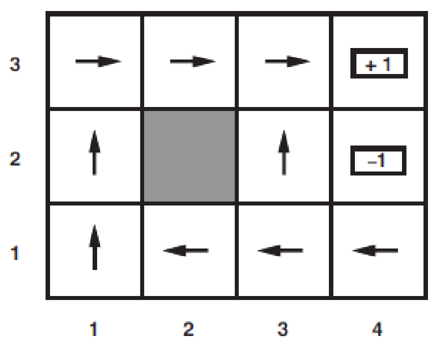
Iteración de Política

Como implementar POLICY-EVALUATION:

- No se necesita resolver las ecuaciones de Bellman tradicionales, ya que las acciones están fijadas por la política
- En iteración i la política π_i especifica la acción $\pi_i(s)$ en estado s
  Esto significa que la utilidad de s a partir de sus vecinos es:

$$U_i(s) = R(s) + \gamma \sum_{s'} P(s' \mid s, \pi_i(s)) U_i(s')$$

- Para el ejemplo, se tiene las ecuaciones:



$$\begin{aligned}
U_i(1,1) &= -0.04 + 0.8U_i(1,2) + 0.1U_i(1,1) + 0.1U_i(2,1)\ , \\
U_i(1,2) &= -0.04 + 0.8U_i(1,3) + 0.2U_i(1,2)\ , \\
U_i(2,1) &= \\
&\vdots
\end{aligned}$$

Iteración de Política

Como implementar POLICY-EVALUATION:

- El conjunto de ecuaciones se puede resolver con algebra lineal en $O(n^3)$, aunque puede ser prohibitivo para n grande
- Alternativamente, se puede usar una versión iterativa simplificada de la actualización de Bellman:

$$U_{i+1}(s) \leftarrow R(s) + \gamma \sum_{s'} P(s' \mid s, \pi_i(s)) U_i(s')$$

Iteración de Política

Algoritmo:

function POLICY-ITERATION(mdp) **returns** a policy
  inputs: mdp, an MDP with states S, actions $A(s)$, transition model $P(s' \mid s, a)$
  local variables: U, a vector of utilities for states in S, initially zero
    π, a policy vector indexed by state, initially random

  repeat
    $U \leftarrow$ POLICY-EVALUATION(π, U, mdp)
    $unchanged? \leftarrow$ true
    for each state s **in** S **do**
      if $\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\ U[s'] > \sum_{s'} P(s' \mid s, \pi[s])\ U[s']$ **then do**
        $\pi[s] \leftarrow \operatorname{argmax}_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\ U[s']$
        $unchanged? \leftarrow$ false
  until $unchanged?$
  return π

Material complementar

- https://www.youtube.com/watch?v=gzFN6UTTpH0&t=178s
- https://www.youtube.com/watch?v=pljiFgRnBAQ
- https://www.youtube.com/watch?v=ZoRMKs8XLSA
- https://www.analyticsvidhya.com/blog/2017/01/introduction-to-reinforcement-learning-implementation/

Preguntas?