

Parte II

Aprendizaje por Refuerzo

Aprendizaje por Refuerzo

Aprendizaje por Refuerzo

Máster en Ciencia y Tecnología Informática

Fernando Fernández Rebollo

Grupo de Planificación y Aprendizaje (PLG)
Departamento de Informática
Escuela Politécnica Superior
Universidad Carlos III de Madrid

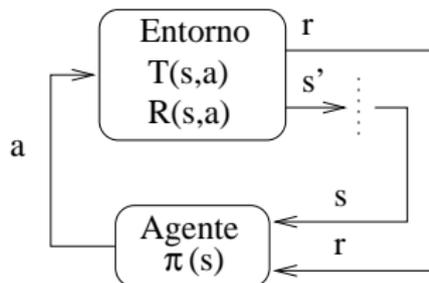


En Esta Sección:

- 3 Aprendizaje por Refuerzo
 - Aprendizaje Por Refuerzo
 - Aproximaciones Libres de Modelo
 - Métodos Basados en el Modelo
 - Representación de la función Q
- 4 Generalización en Aprendizaje por Refuerzo
- 5 Aplicaciones del Aprendizaje por Refuerzo

Introducción

- Problema de Aprendizaje por Refuerzo (definido como un MDP):
 - Conjunto de todos los posibles estados, \mathcal{S} ,
 - Conjunto de todas las posibles acciones, \mathcal{A} ,
 - Función de transición de estados desconocida,
 $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathcal{R}$
 - Función de refuerzo desconocida,
 $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{R}$.
- Objetivo: aprender la política de acción $\Pi : \mathcal{S} \rightarrow \mathcal{A}$ que maximice el refuerzo medio esparado.



Métodos Monte Carlo

- Objetivo: estimar Q^* (¿por qué no es suficiente estimar V^* ?)
- Método basado en:
 - alternar la evaluación de política y su mejora
 - la ejecución de episodios de aprendizaje
 - La actualización de Q basada en la media de los refuerzos obtenidos en los distintos episodios

Monte Carlo con Arranque Exploratorio

Monte Carlo ES

- Inicializar, para todo $s \in \mathcal{S}$, $a \in \mathcal{A}$:
 - $Q(s, a) \leftarrow$ valor arbitrario
 - $\pi(s) \leftarrow$ valor arbitrario
 - $ganancias(s, a) \leftarrow$ lista vacía
- Repetir para siempre
 - 1 Generar un episodio usando arranque exploratorio y π
 - 2 Para cada par (s, a) que aparece en el episodio
 - $R \leftarrow$ ganancia obtenida tras la primera ocurrencia del par (s, a)
 - Añadir R a $ganancias(s, a)$
 - $Q(s, a) \leftarrow promedio(ganancias(s, a))$
 - 3 Para cada s en el episodio
 $\pi(s) \leftarrow \arg \max_a Q(s, a)$

Métodos de Diferencia Temporal (TD)

- Combinación de las ideas de la Programación Dinámica y los métodos Monte Carlo:
 - Aprendizaje por prueba y error
 - Basado en el cálculo de las funciones de valor-acción
 - Estimaciones calculadas sobre estimaciones
- Algoritmos:
 - Sarsa: *on-policy*
 - Q-Learning: *off-policy*

SARSA

- Aprendizaje por prueba y error: Método *On-policy*

SARSA (γ, α).

Inicializar $Q(s, a)$, $\forall s \in \mathcal{S}, a \in \mathcal{A}$

Repetir (para cada episodio)

 Inicializa el estado inicial, s , aleatoriamente.

 Selecciona una acción a usando una política derivada de Q y ejecútala

 Repetir (para cada paso del episodio)

 Recibe el estado actual s' , y el refuerzo, r

 Selecciona una acción a' usando una política derivada de Q y ejecútala

$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha[r + \gamma Q(s', a')]$

 Asigna $s \leftarrow s'$

Devuelve $Q(s, a)$

Exploración vs. Explotación

- Métodos de balancear la exploración/explotación
 - Estrategias de selección de acciones:
 - ϵ -greedy:
Ejecuta $\arg_a \max Q(s, a)$ con probabilidad ϵ
Ejecuta una acción aleatoria con probabilidad $1 - \epsilon$
 - Softmax:

$$P(a_i) = \frac{e^{Q(s, a_i)/\tau}}{\sum_{a_j \in \mathcal{A}} e^{Q(s, a_j)/\tau}}$$

- Inicialización de la función Q
- Sesgar la selección de acciones con conocimiento del dominio adicional

Q-Learning (Watkins, 1989)

- Aprendizaje por prueba y error: método *Off-policy*

Q-Learning (γ, α).

Inicializar $Q(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}$

Repetir (para cada episodio)

 Inicializa el estado inicial, s , aleatoriamente.

 Repetir (para cada paso del episodio)

 Selecciona una acción a y ejecútala

 Recibe el estado actual s' , y el refuerzo, r

$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a')]$

 Asigna $s \leftarrow s'$

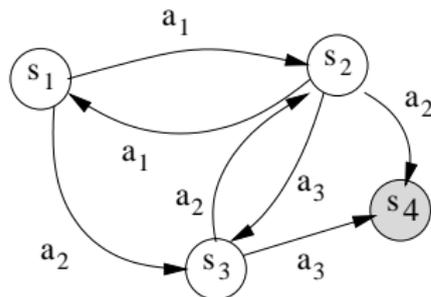
Devuelve $Q(s, a)$

Funciones de Actualización

- Función de actualización determinista:
$$Q(s, a) \leftarrow r + \gamma \max_{a'} Q(s', a')$$
- Función de actualización no determinista:
$$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a')]$$

Ejemplo

- Suponer el siguiente MDP determinista



- Tabla Q Inicial:

$Q(s,a)$	a_1	a_2	a_3
s_1	0	0	0
s_2	0	0	0
s_3	0	0	0
s_4	0	0	0

Ejemplo

- El agente ejecuta el siguiente episodio o secuencia de acciones: $s_1 \rightarrow^{a_1} s_2 \rightarrow^{a_3} s_3 \rightarrow^{a_3} s_4$
- Actualizaciones en la tabla Q:
 - $Q(s_1, a_1) = \mathcal{R}(s_1, a_1) + \gamma \arg_a \max Q(s_2, a) = 0 + \gamma 0 = 0$
 - $Q(s_2, a_3) = \mathcal{R}(s_2, a_3) + \gamma \arg_a \max Q(s_3, a) = 0 + \gamma 0 = 0$
 - $Q(s_3, a_3) = \mathcal{R}(s_3, a_3) + \gamma \arg_a \max Q(s_4, a) = 1 + \gamma 0 = 1$
- Tabla Q resultante:

$Q(s,a)$	a_1	a_2	a_3
s_1	0	0	0
s_2	0	0	0
s_3	0	0	1
s_4	0	0	0

Ejemplo

- Segundo episodio de aprendizaje: $s_1 \xrightarrow{a_2} s_3 \xrightarrow{a_2} s_2 \xrightarrow{a_2} s_4$
- Actualizaciones en la tabla Q:
 - $Q(s_1, a_2) = \mathcal{R}(s_1, a_2) + \gamma \arg_a \max Q(s_3, a) = 0 + \gamma \max(0, 0, 1) = \gamma = 0,5$
 - $Q(s_3, a_2) = \mathcal{R}(s_3, a_2) + \gamma \arg_a \max Q(s_2, a) = 0 + \gamma 0 = 0$
 - $Q(s_2, a_2) = \mathcal{R}(s_2, a_2) + \gamma \arg_a \max Q(s_4, a) = 1 + \gamma 0 = 1$
- Tabla Q resultante:

Q(s,a)	a ₁	a ₂	a ₃
s ₁	0	0,5	0
s ₂	0	1	0
s ₃	0	0	1
s ₄	0	0	0

Ejemplo

- Tabla Q óptima:

$Q^*(s, a)$	a_1	a_2	a_3
s_1	0,5	0,5	0,25
s_2	0,25	1	0,5
s_3	0,5	0,5	1
s_4	0	0	0

- Política óptima:
 - $\pi^*(s_3) = \arg_a \max Q(s_3, a) = a_3$
 - $\pi^*(s_2) = a_2$
 - $\pi^*(s_1) = a_1$
- Otra política óptima: igual que la anterior pero con $\pi^*(s_1) = a_2$

Métodos Basados en el Modelo para Resolver MDP's

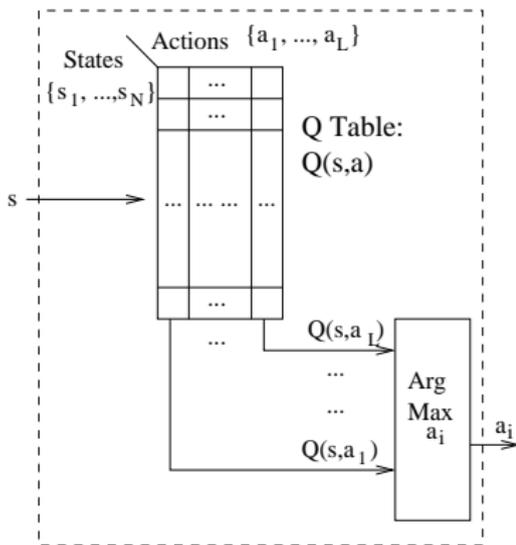
- Si no se conoce el modelo: aprenderlo
- Ejemplo: Dyna-Q
 - Similar a Q-Learning
 - En cada paso, también actualiza su conocimiento del modelo
 - El modelo es utilizado para realizar nuevas actualizaciones de Q

Métodos Basados en el Modelo: Dyna-Q

Algoritmo Dyna-Q

- Inicializar $Q(s, a)$ y $\text{Modelo}(s, a)$ arbitrariamente
- Repetir para siempre
 - Inicializar s
 - Seleccionar una acción a a partir de s usando una política derivada de Q
 - $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$
 - $\text{Modelo}(s, a) \leftarrow s', r$
 - $s \leftarrow s'$
 - Repetir N veces
 - $s \leftarrow$ estado visitado anteriormente y elegido aleatoriamente
 - $a \leftarrow$ acción aleatoria ejecutada anteriormente desde s
 - $s', r \leftarrow \text{Modelo}(s, a)$
 - $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$

Representación Tabular de la Función Q



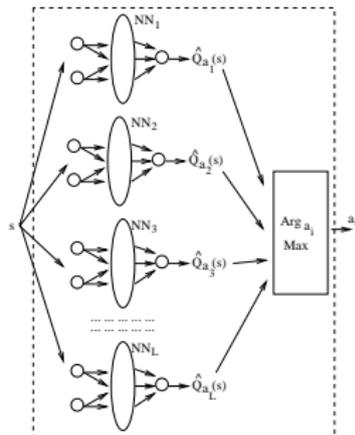
- Problema: espacio de estados continuo o de gran tamaño
- Solución: métodos de generalización
 - Aproximaciones ad-hoc basadas en conocimiento del dominio
 - Discretización del espacio de estados
 - Aproximación de funciones

En Esta Sección:

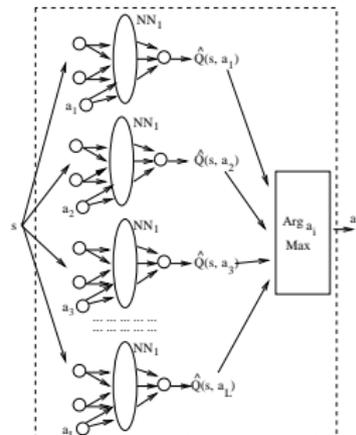
- 3 Aprendizaje por Refuerzo
- 4 Generalización en Aprendizaje por Refuerzo
 - Aproximación de Funciones
- 5 Aplicaciones del Aprendizaje por Refuerzo

Aproximación de Funciones

- Utilizar un aproximador de funciones para representar la función Q :



L aproximadores ($\hat{Q}_{a_i}(s)$)



1 aproximador ($\hat{Q}(s, a)$)

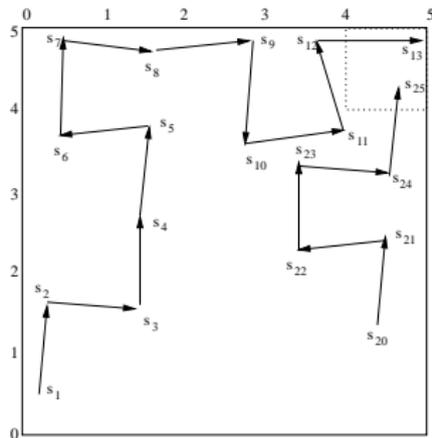
Batch Q-Learning (1/2)

- Entradas:
 - 1 Un espacio de estados X
 - 2 Un conjunto de L acciones, $A = \{a_1, \dots, a_L\}$
 - 3 Una colección T de N tuplas de experiencia del tipo $\langle s, a_i, s', r \rangle$, donde $s \in X$ es un estado desde donde la acción a_i es ejecutada y s' es el estado resultante

Batch Q-Learning (2/2)

- Sea $\hat{Q}^0(s, a) = 0$
- $iter = 0$
- Repetir
 - Inicializar los conjuntos de aprendizaje $\mathcal{T}^{iter} = \emptyset$
 - Desde $j=1$ hasta N , utilizando la j ésima tupla $\langle s_j, a_j, s'_j, r_j \rangle$ > hacer
 - $c_j = r_j + \max_{a \in A} \gamma \hat{Q}^{iter-1}(s'_j, a)$
 - $\mathcal{T}^{iter} = \mathcal{T}^{iter} \cup \{ \langle s_j, a_j, c_j \rangle \}$
 - Entrenar $\hat{Q}^{iter}(s, a)$ para aproximar el conjunto de aprendizaje \mathcal{T}^{iter}
 - $iter = iter + 1$
- Hasta que el vector c_j no cambie

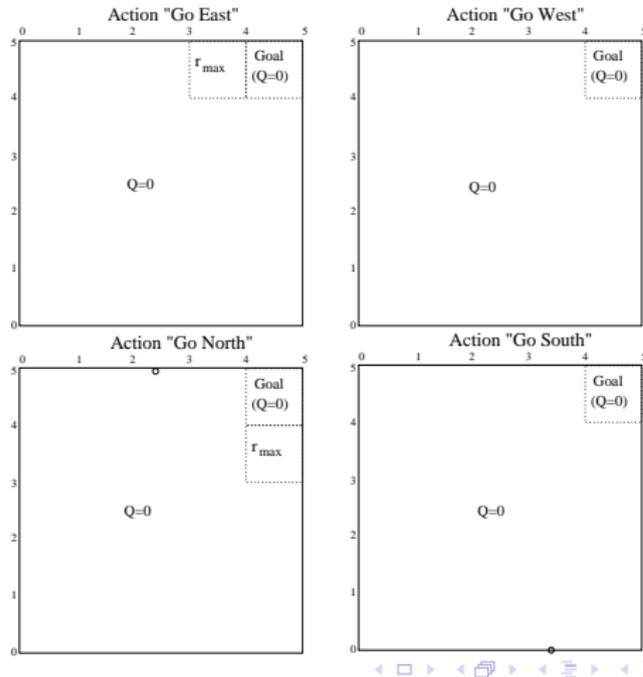
Generar Tuplas de Experiencia



- $T^0 = \{(s_1, \text{norte}, 0), (s_2, \text{este}, 0), (s_3, \text{norte}, 0), (s_4, \text{norte}, 0), \dots, (s_{12}, \text{este}, r_{\max}), \dots, (s_{20}, \text{norte}, 0), (s_{21}, \text{este}, 0), \dots, (s_{24}, \text{norte}, r_{\max}), \dots\}$

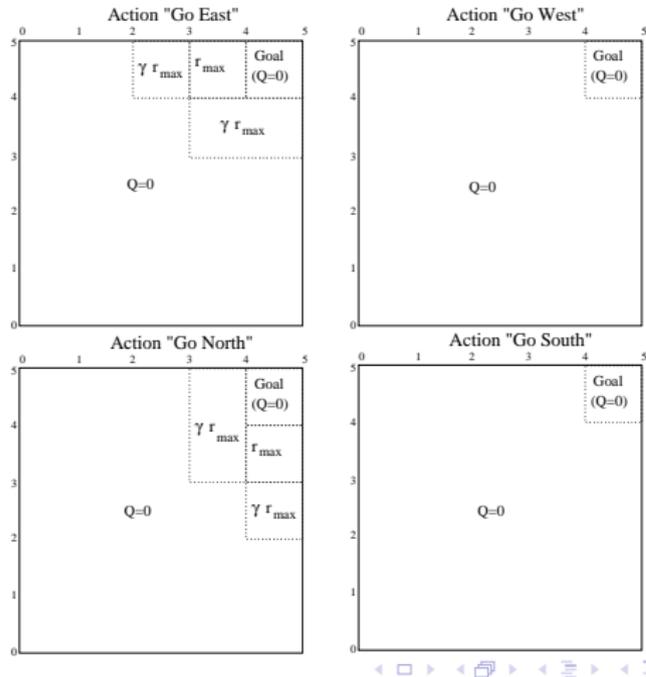
Batch Q-Learning: Iter. 1

$$Q^1(s, a):$$



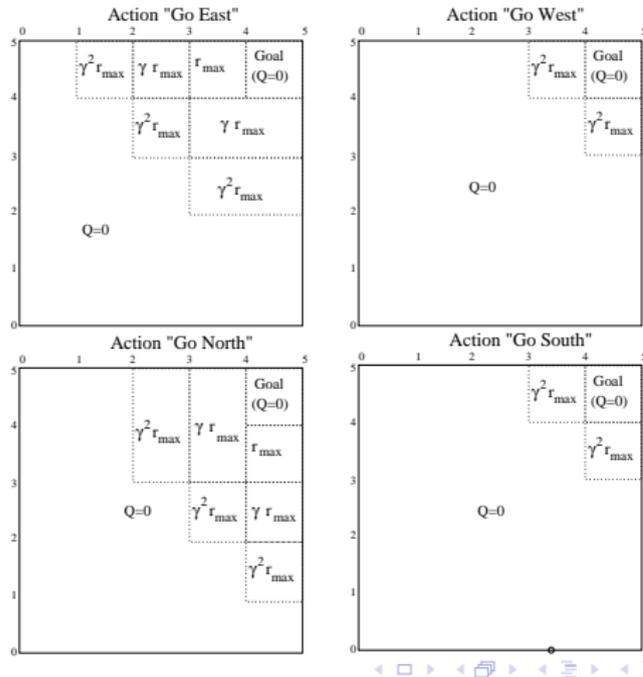
Batch Q-Learning: Iter. 2

$$Q^2(s, a):$$



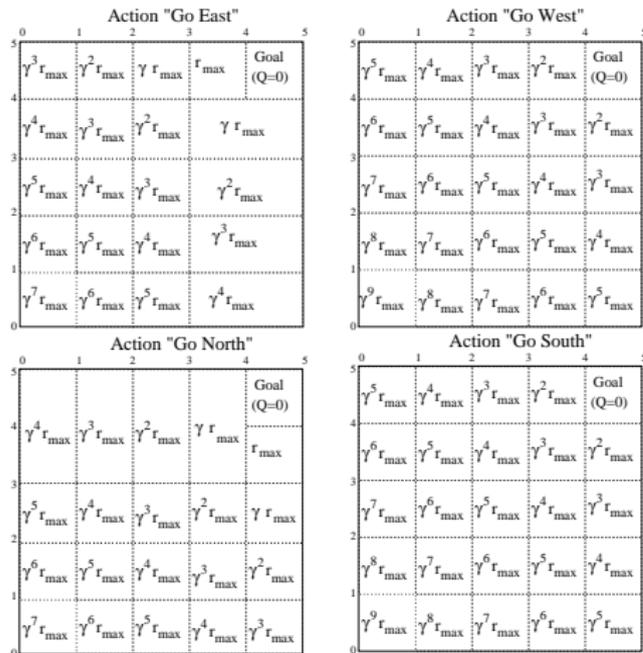
Batch Q-Learning: Iter. 3

$$Q^3(s, a):$$



Ejecución de Batch Q-Learning: Iteración 9

$$Q^9(s, a):$$



Métodos Actor-crítico

- Cuando el espacio de acciones también es continuo, la acción puede ser parte de la entrada del aproximador
- Inconveniente: recuperar $\max_a Q(s, a)$
- Solución:
 - Usar un aproximador para la política, $\pi(s)$ (actor)
 - Usar un aproximador para la función de valor, $Q(s, a)$ (crítico)

Aprendizaje con Aproximación de Funciones

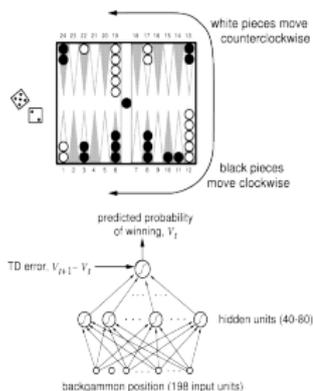
- Problemas en la aplicación de aprendizaje supervisado en aprendizaje por refuerzo:
 - Las etiquetas de los datos (valor Q) son desconocidos a priori
 - Los refuerzos positivos pueden ser escasos comparados con refuerzos nulos (problema de clasificación de conjuntos no balanceados)
 - Los distintos pasos del aprendizaje pueden tener distintas características de aprendizaje
 - Las estimaciones son calculadas sobre estimaciones:
propagación de errores:
 - Buena convergencia: se obtienen la función de valor y política óptimas
 - Convergencia por casualidad: la función de valor calculada es subóptima pero genera una política óptima
 - Mala convergencia: tanto la función de valor como la política son subóptimas
 - Divergencia: no se converge ni a una función de valor ni a una

En Esta Sección:

- 3 Aprendizaje por Refuerzo
- 4 Generalización en Aprendizaje por Refuerzo
- 5 Aplicaciones del Aprendizaje por Refuerzo
 - Ejemplos de Aplicación

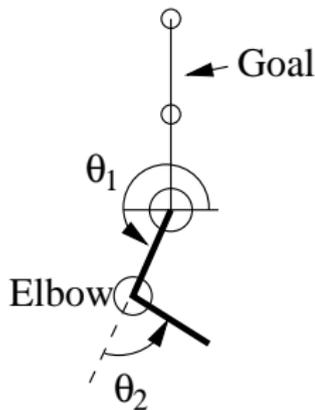
TD-Gammon

- Juego del backgammon
 - 34 piezas y 24 posiciones posibles: espacio de estados enorme!!!
 - 20 posibles movimientos por cada tirada de dado
 - Perfecto concimiento de espacios de estado, acción, y transiciones de estado
 - Refuerzo cuando se gana la partida
 - $V(s)$ evalúa la probabilidad de ganar desde el estado s
 - Uso de una red de neuronas para aproximar $V(s)$
 - Aprendizaje mediante una versión no lineal de TD(λ)
 - Aprendizaje de los pesos mediante descenso de gradiente



Problemas de control: Acrobot

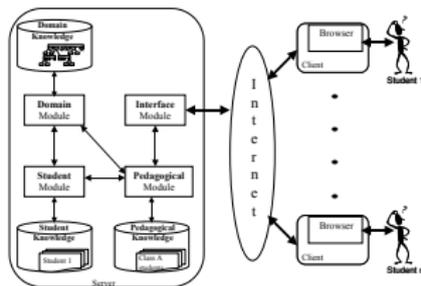
- Acrobot
 - Espacio de estados continuo: 2 ángulos y 2 velocidades
 - Espacio de acciones discreto: 2 acciones: empujar el codo en una dirección y otra
 - Objetivo: colocar el brazo en posición vertical invertida
 - Perfecto concimiento de espacios de estado, acción, y transiciones de estado
 - Diversas aproximaciones para generalización: *Variable Resolution Discretization*, redes de neuronas, árboles de decisión, etc.



Módulo Pedagógico en Tutores Inteligentes

● RLATES:

- Estado: conocimiento que dispone el alumno sobre el contenido del tutor
- Acción: mostrar un contenido al alumno
- Percepción realizada mediante tests
- Objetivo: obtener una política pedagógica
- Simplificación a un espacio de estados con características binarias
- Difícil percepción/modelización del estado: POMDP



Módulo Pedagógico en Tutores Inteligentes

The screenshot shows a web browser window displaying a tutorial page titled "E-R BASICO". The page is part of a course "Curso de Diseño de Bases de Datos Asistido por Ordenador". The interface includes a navigation menu on the left with categories like "INTERRELACION", "ATRIBUTO DE INTERRELACION", "TIPOS DE CORRESPONDENCIA", "CARDINALIDAD", "GRADO", "ENTIDAD", and "ATRIBUTO". The main content area features a "Definition" tab, a "Definition" section with a book icon, and a text block explaining the E-R model. Below the text, there is a "LIBRO" button and a diagram of an entity-attribute relationship.

Curso de Diseño de Bases de Datos Asistido por Ordenador

E-R BASICO

Anterior Salir Siguiente

Definición Tests

Definición:

El **modelo E/R básico** es aquella parte del modelo E/R, que contiene los conceptos, reglas y convenciones que permiten representar el universo del discurso, recogiendo la semántica básica del mismo.

Los elementos que comprende este modelo son:

- Entidad.

LIBRO

ENTIDAD — Nombre del atributo

Otras Aplicaciones

- Adaptive Cognitive Orthotics: Combining Reinforcement Learning and Constraint-Based Temporal Reasoning by Matthew Rudary, Satinder Singh and Martha Pollack. In Proceedings of the Twenty-First International Conference on Machine Learning (ICML), pages 719-726, 2004.
- Cobot: A Social Reinforcement Learning Agent by Charles Isbell, Christian Shelton, Michael Kearns, Satinder Singh and Peter Stone. In Advances in Neural Information Processing Systems 14 (NIPS) pages 1393-1400, 2002.
- Empirical Evaluation of a Reinforcement Learning Spoken Dialogue System by Satinder Singh, Michael Kearns, Diane Litman, and Marilyn Walker. In Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI), pages 645-651, 2000
- Mori, T., Nakamura, Y., Sato, M., and Ishii, S. (2004). Reinforcement learning for CPG-driven biped robot. Nineteenth National Conference on Artificial Intelligence (AAAI), pp.623-630.