Aprendizaje por Refuerzo

Aprendizaje Automático

Ingeniería Informática

Fernando Fernández Rebollo y Daniel Borrajo Millán

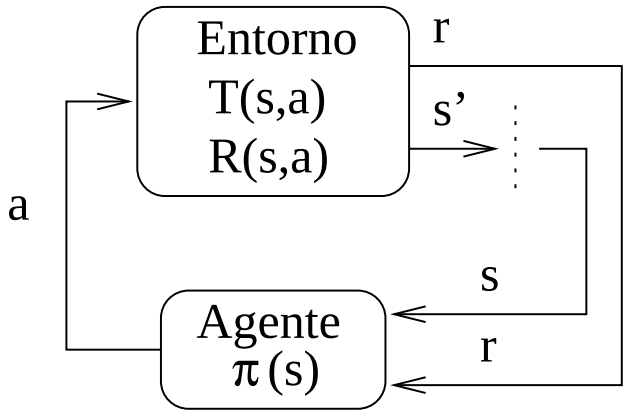
Grupo de Planificación y Aprendizaje (PLG)
Departamento de Informática
Escuela Politécnica Superior
Universidad Carlos III de Madrid

27 de febrero de 2009

En Esta Sección:

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, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \Re$
  - Función de refuerzo desconocida, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \Re$.



- Objetivo: aprender la política de acción $\Pi : \mathcal{S} \rightarrow \mathcal{A}$ que maximice el refuerzo medio esparado.

Q-Learning (Watkins, 1989)

- No se conocen las funciones de transición de estado ni de refuerzo
- Aprendizaje por prueba y error

Q-Learning (γ, α).

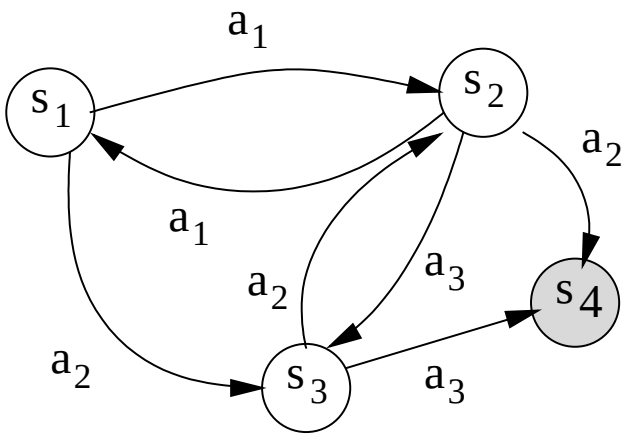
Inicializar $Q(s, a)$, $\forall s \in \mathcal{S}, a \in \mathcal{A}$
Repetir (para cada episodio)
    Inicializa el estado inicial, s, aleatoriamente.
    Repetir (para cada paso del episodio)
        Selecciona una acción a y ejecútala
        Recibe el estado actual s', y el refuerzo, r
        $Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha[r + \gamma \text{ máx}_{a'} Q(s', a')]$
        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: $s_1 \rightarrow^{a_2} s_3 \rightarrow^{a_2} s_2 \rightarrow^{a_2} s_4$
- 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 $\pi^*(s_1) = a_2$

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

Métodos Basados en el Modelo para Resolver MDP's

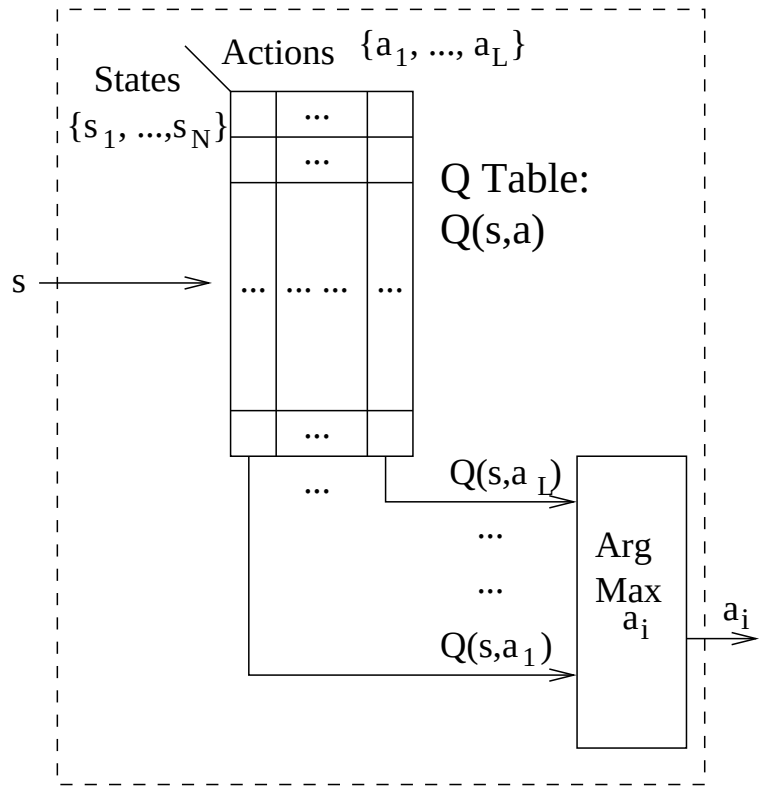
- Si no se conoce el modelo: aprenderlo
- Ejemplo: Dyna-Q
  - Similar a Q-Learning
  - En cada paso, también actualiza su conocimiento del modelo
  - El modelo es utilizado para realizar nuevas actualizactiones de Q

Métodos Basados en el Modelo: Dyna-Q

Algoritmo Dyna-Q

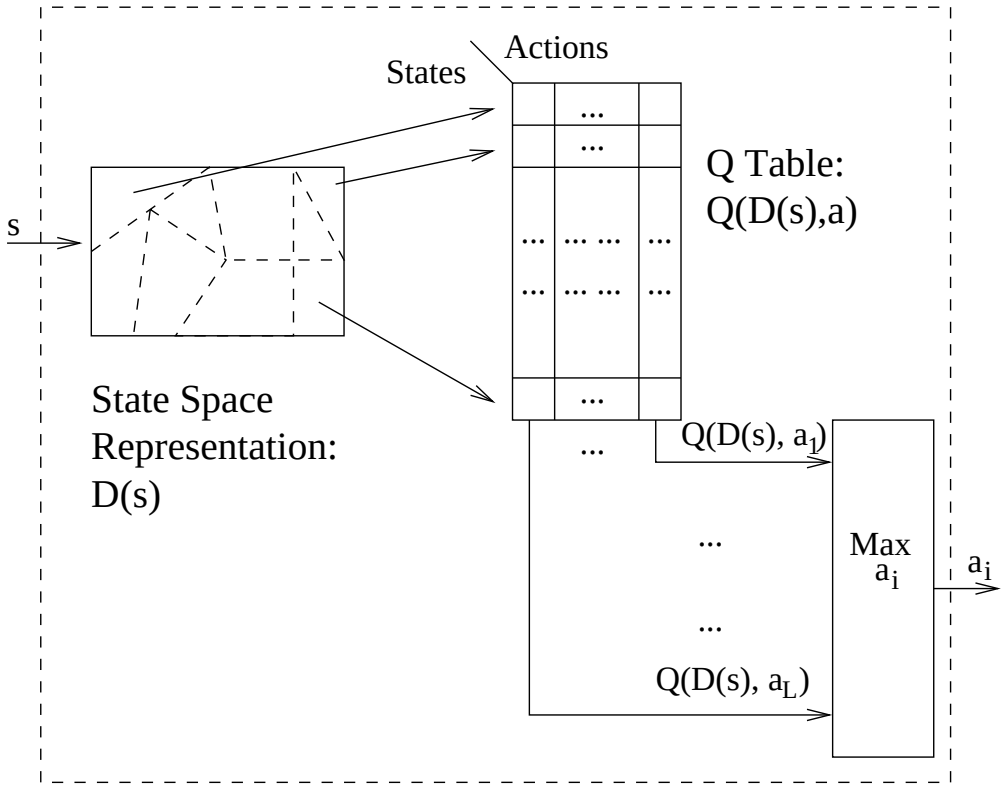
- Inicializar $Q(s, a)$ y Modelo(s, a) arbitrariamente
- Repetir para siempre
  - Inicializar s
  - Seleccionar una acción a a partir de s usando una política derivada de Q
  - $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \text{máx}_{a'} Q(s', a') - Q(s, a)]$
  - Modelo$(s, a) \leftarrow s', r$
 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

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

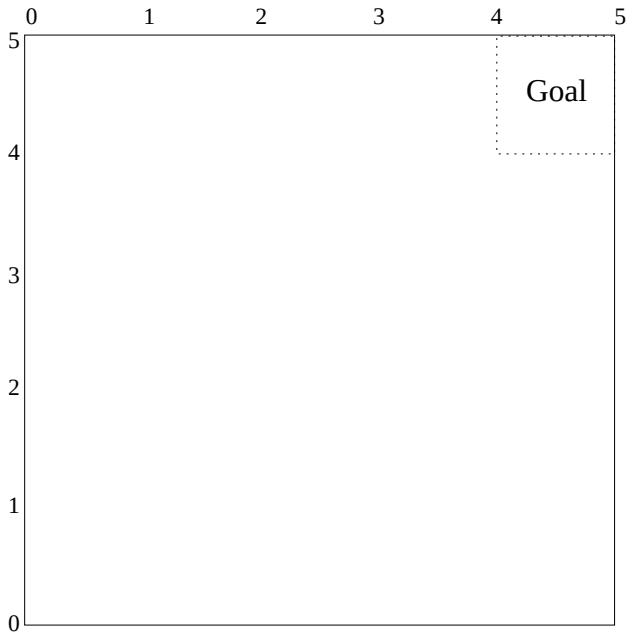
Discretización del Espacio de Estados

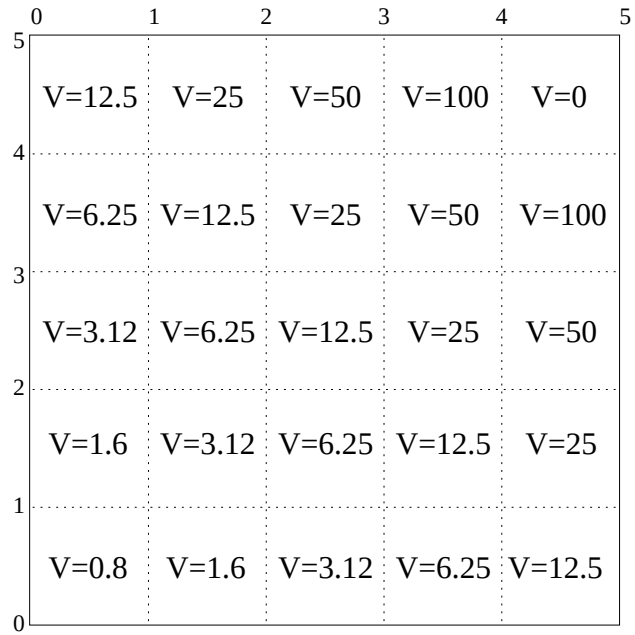


- Problema:
  - Discretizaciones erróneas pueden romper fácilmente la propiedad de Markov
  - Cuántas regiones necesitamos para discretizar el espacio de estados?

Ejemplo

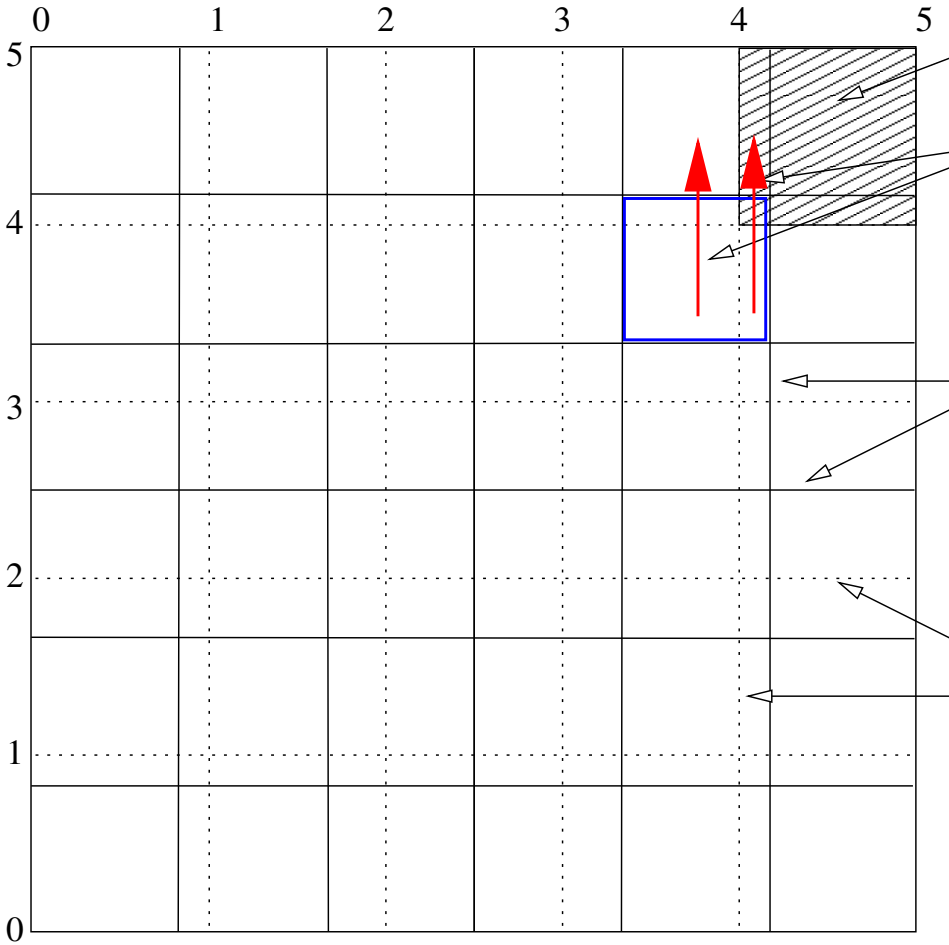
- Dominio de navegación de un robot:
  - Espacio de estados continuo de tamaño 5×5
  - Acciones: Norte, Sur, Este, Oeste, de tamaño 1





- Discretización óptima de tamaño 5×5

Pérdida de la Propiedad de Markov



Goal Area

Non–determinism Introduced:
Same action from same state produces different immediate rewards

Limits of the Regions
(6 x 6 discretization)

Value Function Limits

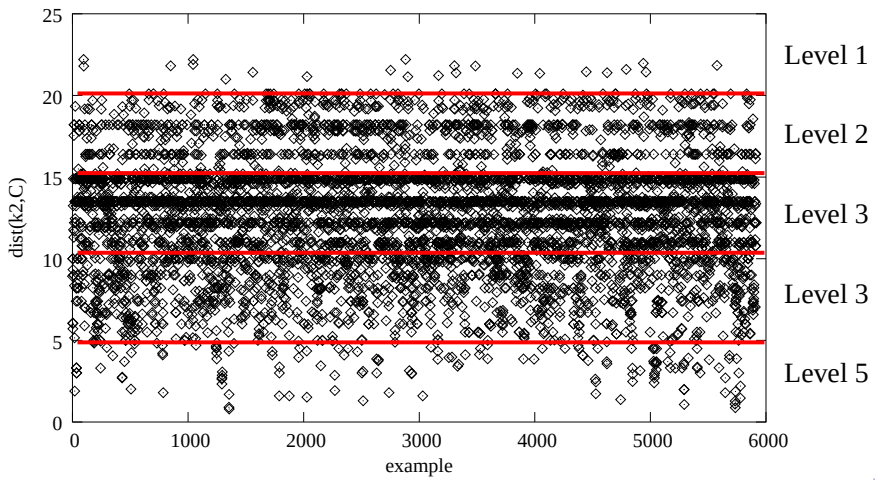
Keepaway (Stone, Sutton and Kuhlmann, 05)

Ejemplo: la Tarea Keepaway

- Espacio de estados: 19 atributos continuos (Los *keepers* y los *takers* se ordenan tomando en cuenta su distancia al jugador)
  - $\text{dist}(k_1, C)$, ..., $\text{dist}(k_4, C)$
  - $\text{dist}(t_1, C)$, ..., $\text{dist}(t_3, C)$
  - $\text{dist}(k_1, t_1)$, ..., $\text{dist}(k_1, t_3)$
  - $\text{Min}(\text{dist}(k_2, t_1), \text{dist}(k_2, t_2), \text{dist}(k_2, t_3))$
  - etc...
- Espacio de estados discreto: 4 acciones
  - Mantener la pelota
  - Pasar a $k2$, Pasar a $k3$, Pasar a $k4$
- Función de transición de estados desconocida
- Función de refuerzo desconocida

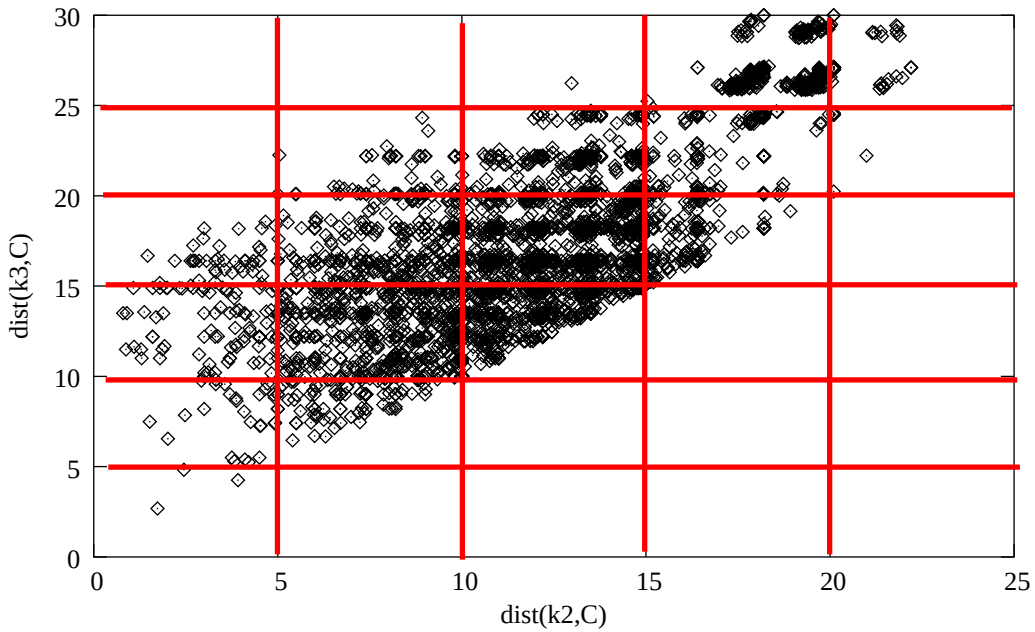
Discretización uniforme del espacio de estados

- Discretizar cada atributo en un número dado de niveles de discretización o regiones
- En *Keepaway*:
  - d=5 niveles de discretización
  - f=19 atributos
  - $d^f = 1{,}907348e + 13$ regiones/estados
- Ejemplo para la característica 2 (dist(k_2, C)):



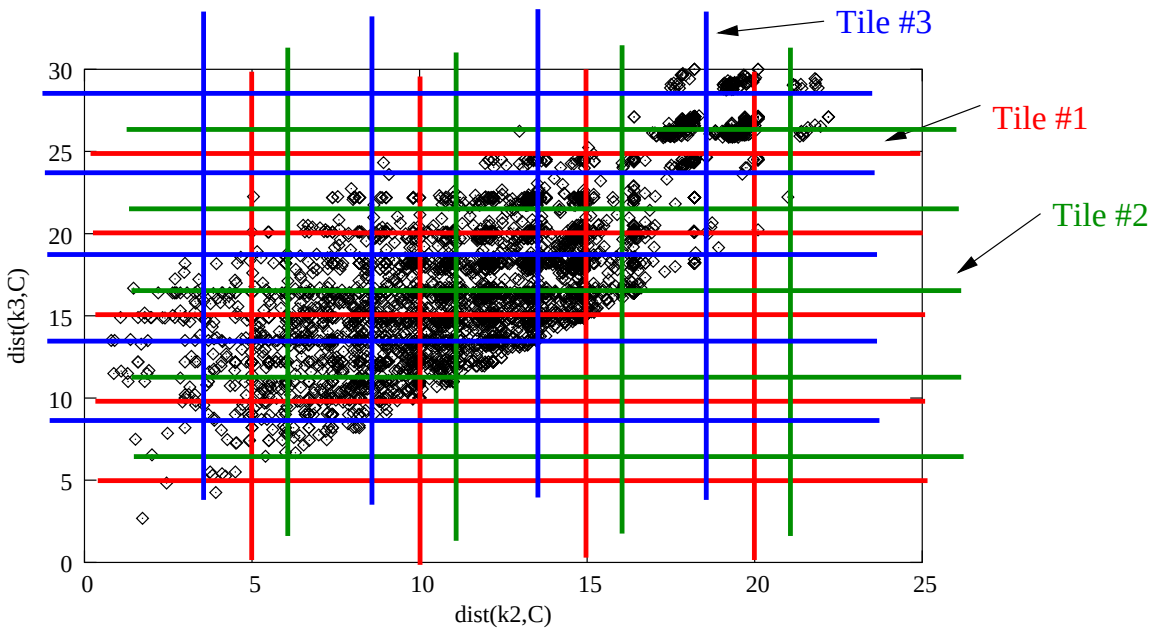
Regiones Generadas por la Discretización Uniforme

- Proyección del espacio de estados sobre los atributos 2 y 3:



CMAC (Albus, 81)

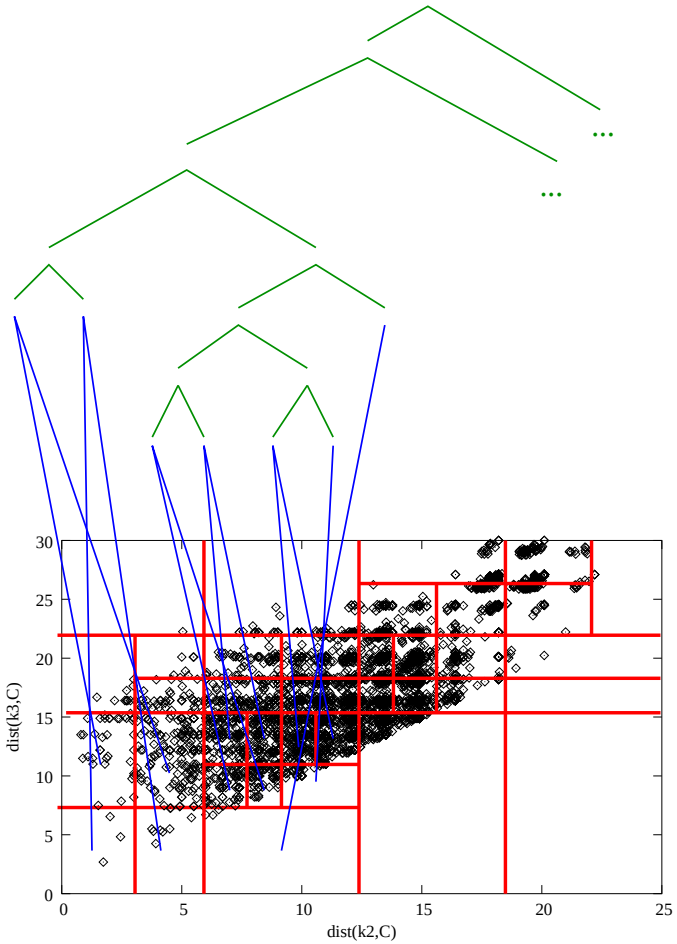
- Superponer varias discretizaciones:



- Cada celda mantiene su propia aproximación de la funcón Q: Celda $\#i$ aproxima $Q_i(s, a)$
- $Q(s, a) = f(Q_1(s, a), Q_2(s, a), Q_3(s, a))$

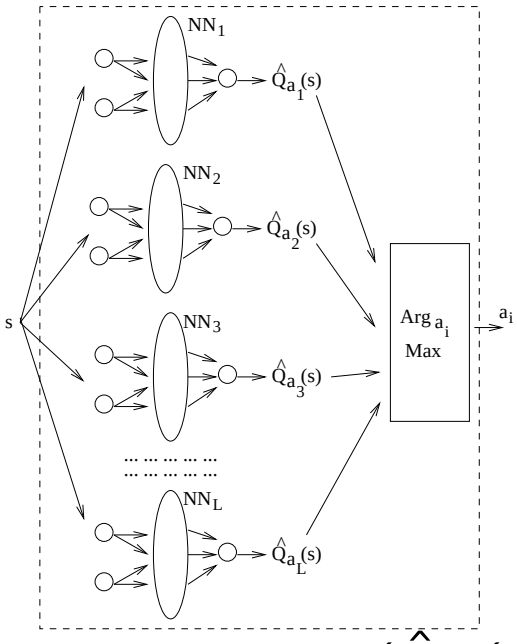
Discretización de Resolución Variable: Árboles KD (Munos and Moore, 02)

- Los nodos y hojas del árbol representan regiones del espacio de estados
- En cada nodo del árbol, una región se divide en dos:
- Los criterios para partir un nodo son diversos, y buscan diferencias dentro de la región:
  - en la función de valor
  - en la política
  - ...

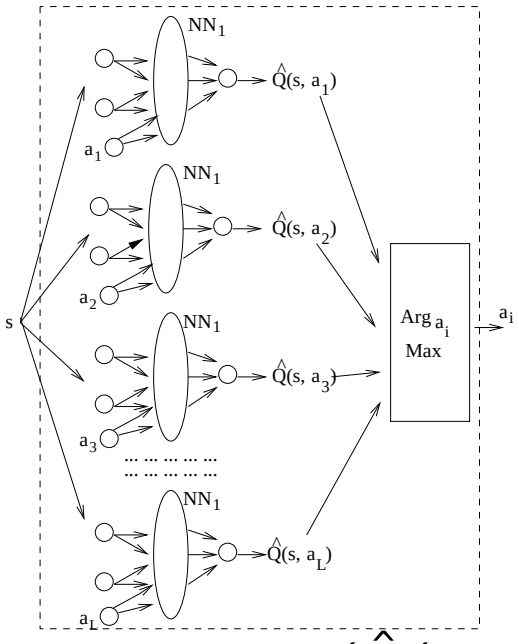


Aproximación de Funciones

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



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



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

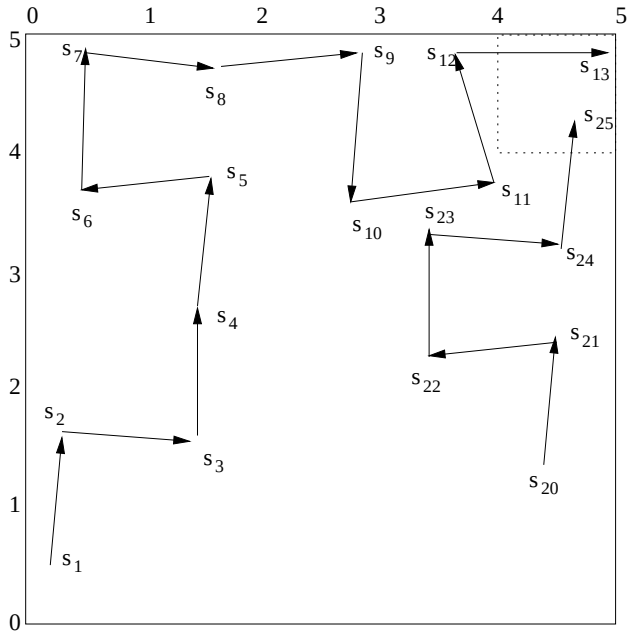
Batch Q-Learning (1/2)

- Entradas:
  1. Un espacio de estados X
  2. Un conjunto de L acciones, $A = \{a_1, \ldots, a_L\}$
  3. Una colección T de N tuplas de experiencia del tipo $< s, a_i, s', r >$, 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 $T^{iter} = \emptyset$
  - Desde j=1 hasta N, utilizando la $j^{\text{ésima}}$ tupla $< s_j, a_j, s'_j, r_j >$ hacer
    - $c_j = r_j + \text{máx}_{a \in A} \gamma \hat{Q}^{iter-1}(s'_j, a)$
    - $T^{iter} = T^{iter} \cup \{< s_j, a_j, c_j >\}$
  - Entrenar $\hat{Q}^{iter}(s,a)$ para aproximar el conjunto de aprendizaje T^{iter}
  - $iter = iter + 1$
- Hasta que el vector c_j no cambie

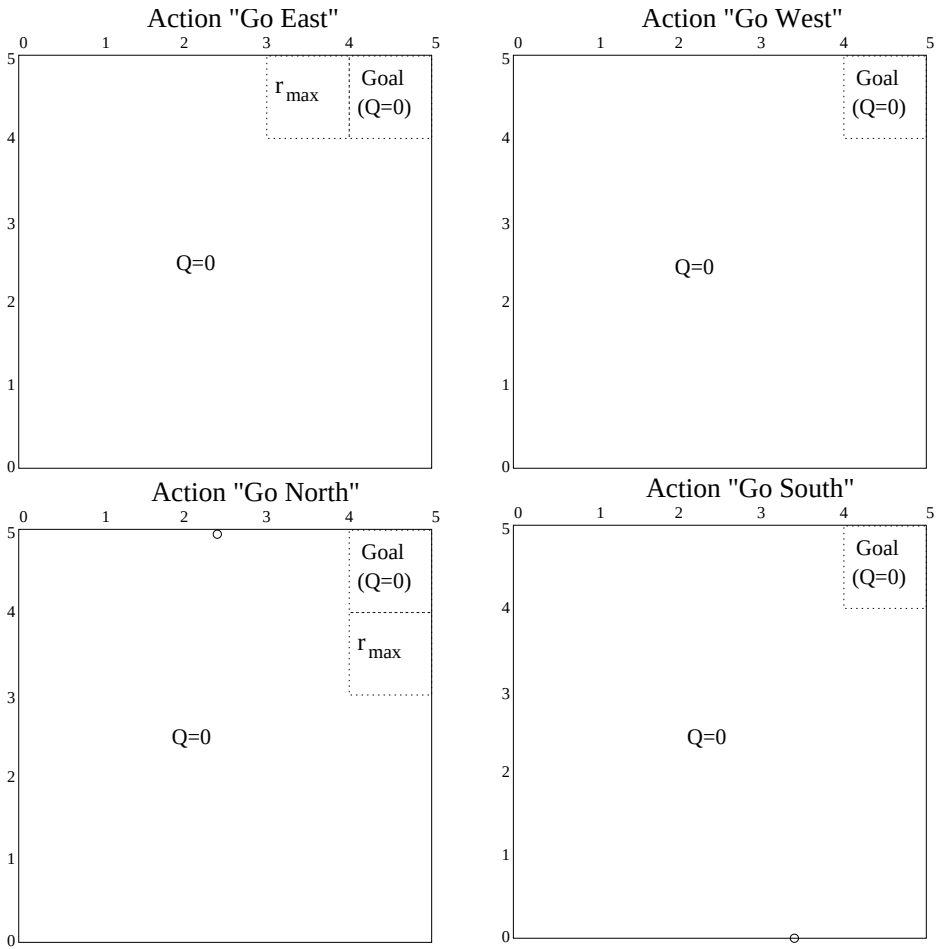
Generar Tuplas de Experiencia



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

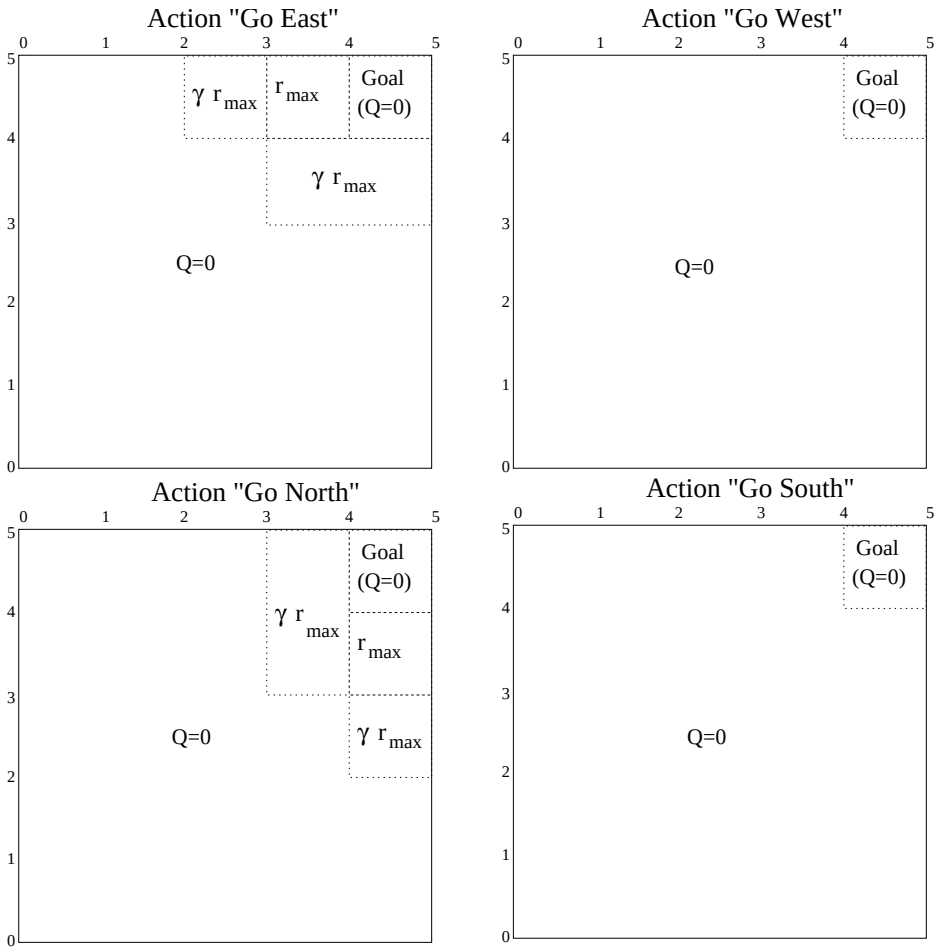
Batch Q-Learning: Iter. 1

$Q^1(s, a)$:



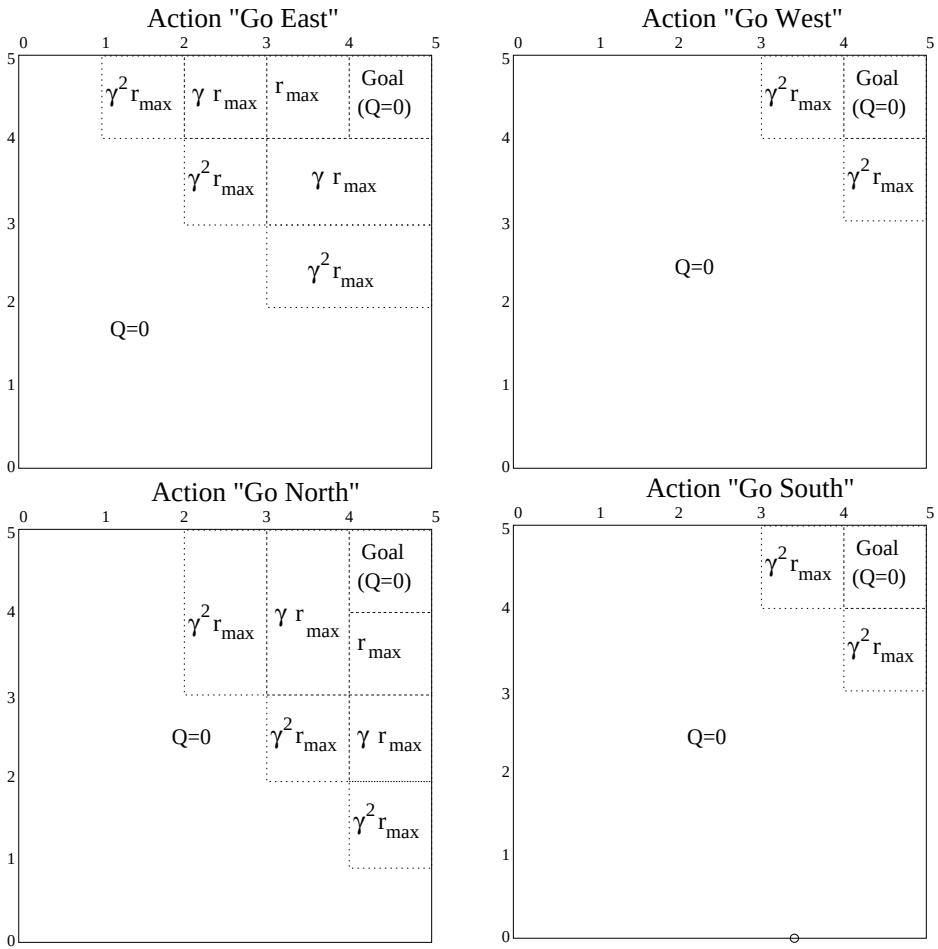
Batch Q-Learning: Iter. 2

$Q^2(s, a)$:



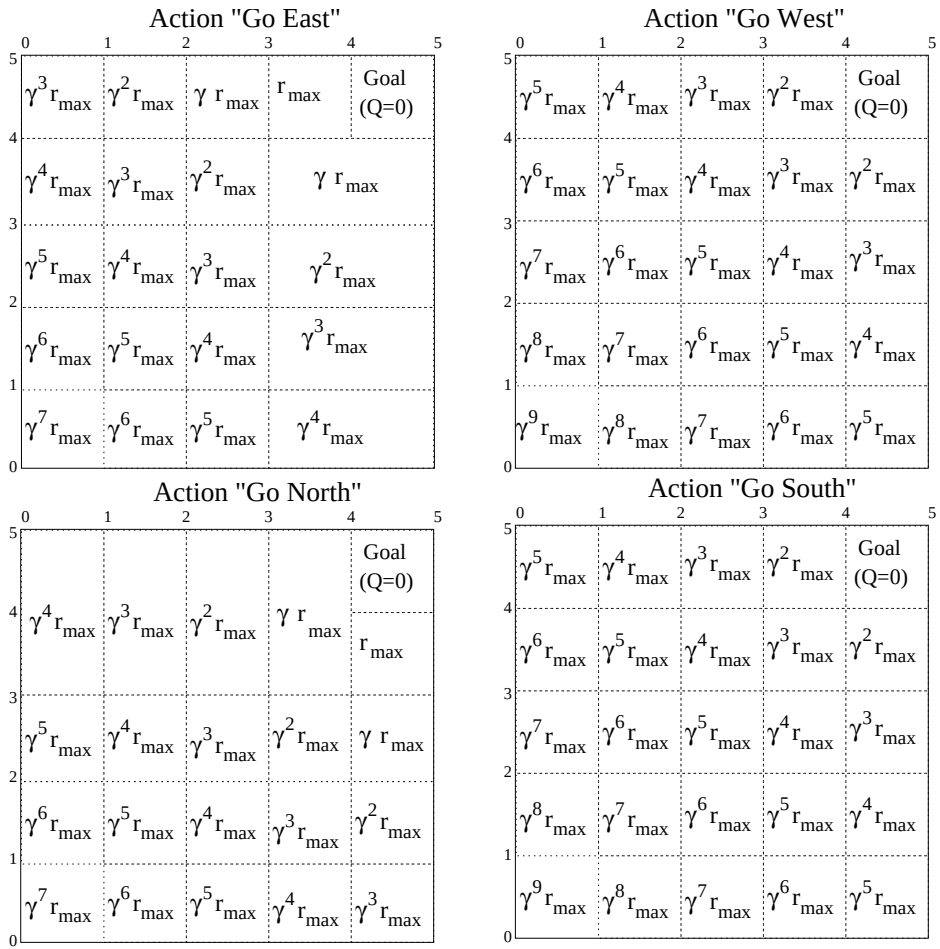
Batch Q-Learning: Iter. 3

$Q^3(s, a)$:



Ejecución de Batch Q-Learning: Iteración 9

$Q^9(s, a)$:

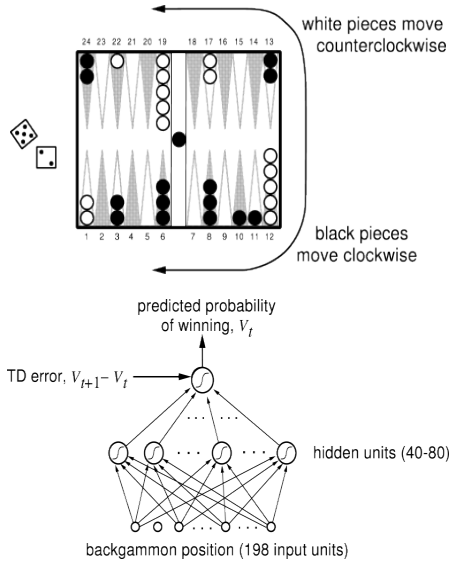


Aprendizaje con Aproximación de Funciones

- 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

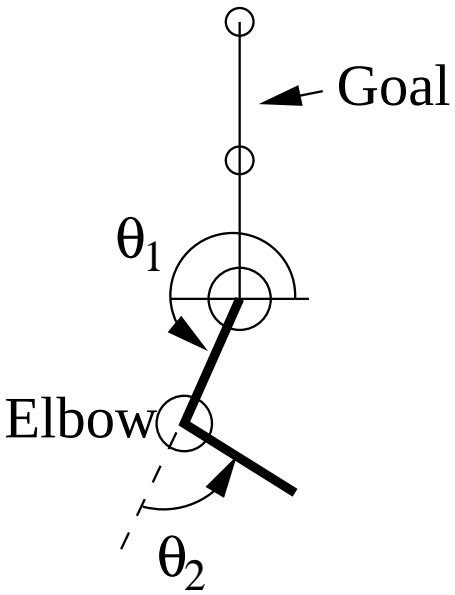
TD-Gammon

- 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) evalua 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



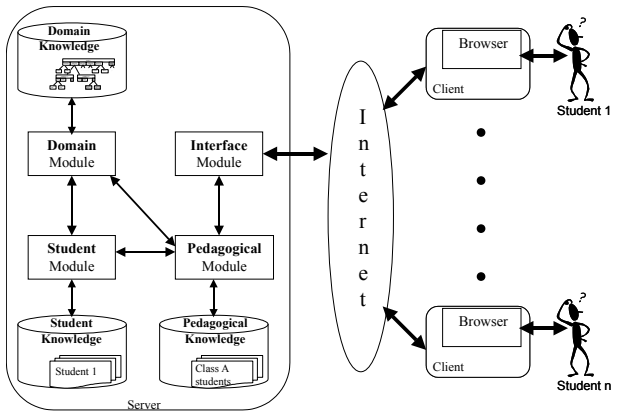
Problemas de control: Acrobot

- 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 posicion 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.

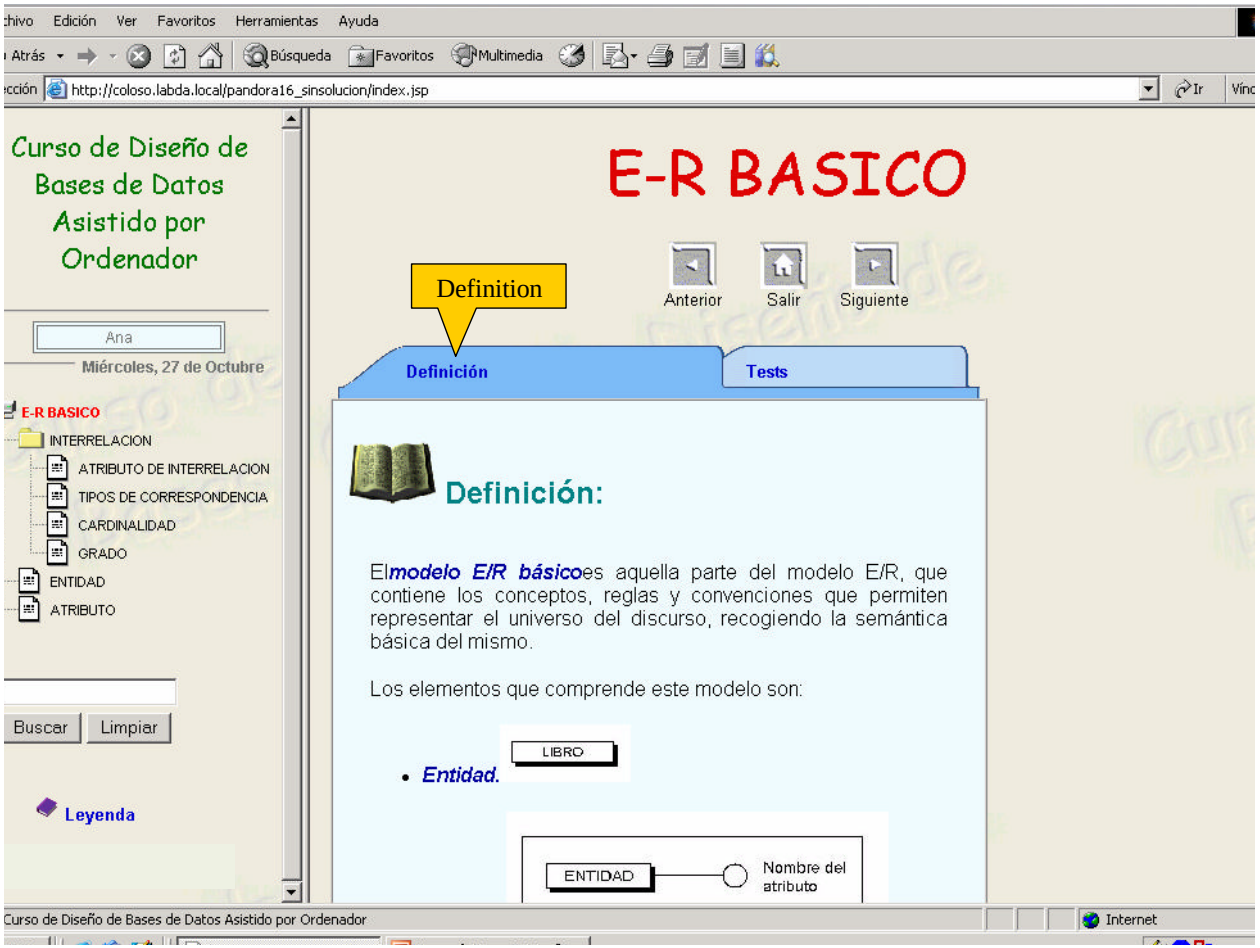


Módulo Pedagógico en Tutores Inteligentes

- 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



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.

Resumen

- Diferencias entre Programación Dinámica, métodos libres de modelo y basados en el modelo
- Planificación en Procesos de Decisión de Markov
- Aprendizaje por Refuerzo:
  - Algoritmo Q-Learning
  - Importancia de la representación de los estados, las acciones y las funciones de valor
  - Discretizaciones incorrectas pueden romper la propiedad de Markov
  - Divergencia de los métodos de aproximación de las funciones de valor

Bibliografía

- Machine Learning, Tom Mitchell. Capítulo 13.
- Reinforcement Learning: An Introduction. Richard Sutton y Andrew Barto. MIT Press. 1998
- Reinforcement Learning Repository: http://www-anw.cs.umass.edu/rlr/
- Aprendizaje Automático: conceptos básicos y avanzados. Basilio Sierra Araujo. Pearson Prentice Hall. 2006