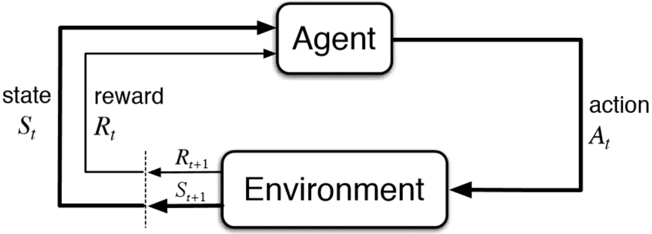
Aprendizaje Automático

uc3m

Aprendizaje por refuerzo

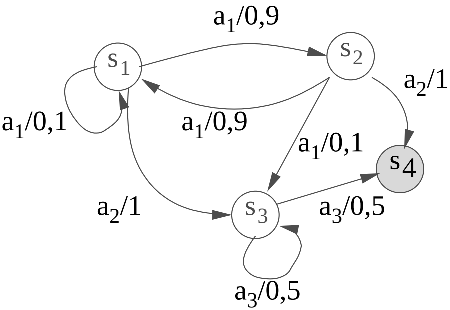
Aprendizaje por Refuerzo

- El aprendizaje por refuerzo consiste en aprender a decidir, ante una situación determinada, qué acción es la más adecuada para lograr un objetivo.
- Proceso iterativo de prueba y error
- Aprendizaje a través de señales de refuerzo



MDP (Markov Decision Process)

- Se asume que el entorno se comporta según un Proceso de Decisión de Markov (MDP) subyacente



Método de Resolución

- Se conoce el modelo
  - Se conoce el modelo (función de transición de estados y función de refuerzo del MDP)
  - Métodos basados en el modelo: Programación Dinámica
- No se conoce el modelo (dos alternativas)
  - Aprender el modelo y usar métodos basados en el modelo, o
  - Aprender las funciones de valor y/o políticas directamente: métodos libres de modelo o Aprendizaje por Refuerzo (Directo)

Aprendizaje Supervisado, No Supervisado y por Refuerzo

- Aprendizaje Supervisado:
  - Aprender $\hat{f} : \vec{x} \rightarrow \vec{y}$. Ejemplos:
    - $\vec{y} \in \{y_1, \ldots, y_n\}$: Clasificación
    - $\vec{y} \in \mathbb{R}$: Regresión
  - A partir de pares $< \vec{x_i}, y_i >$
- Aprendizaje No Supervisado:
  - Aprender $\hat{f} : \vec{x} \rightarrow \vec{y}$. Ejemplos:
    - $\vec{y} \in \{y_1, \ldots, y_n\}$: Clustering
  - A partir de $< \vec{x_i} >$ (sin atributo de clase ni supervisión)
- Aprendizaje por refuerzo:
  - Aprender $\hat{f} : \vec{x} \rightarrow \vec{y}$ donde:
    - $\hat{f}$: política de acción (π)
    - $\vec{x}$: estado o situación en la que se encuentra el agente (s)
    - $\vec{y}$: acción que puede ejecutar el agente (a)
  - A partir de experiencias de interacción con el entorno:
    - Estado, acción, estado siguiente más un valor de refuerzo inmediato $< s_i, a_i, s'_i, r_i >$

En este tema

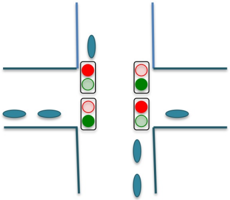
En este tema

Definición de un MDP

- Un MDP se define como una tupla $< S, A, T, R >$, tal que:
  - Cjto de estados S
  - Cjto de acciones A
  - Función de transición
    $T : S \times A \rightarrow P(S)$, $P(S)$ es una distribución de probabilidad sobre S
    $T(s, a, s')$ es la probabilidad de que se realice una transición desde s hasta s' ejecutando la acción a
  - Función de refuerzo
    $R : S \times A \times S \rightarrow \mathbb{R}$, que proporciona el refuerzo recibido al ejecutar la acción a en el estado s y obtener el estado s'

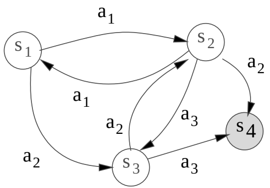
Ejemplo de control de semáforos

- Dados cuatro semáforos en un cruce, encontrar control óptimo
- Estados: estado de cada semáforo, número de coches en cada semáforo
- Acciones: cambiar uno o varios semáforos de rojo a verde o viceversa
- Refuerzo:
  - si posibles cruces: $-\infty$
  - si no: $-$# coches esperando



Ejemplo de MDP Determinista

La ejecución de una acción desde un estado siempre produce la misma transición de estado y el mismo refuerzo/coste

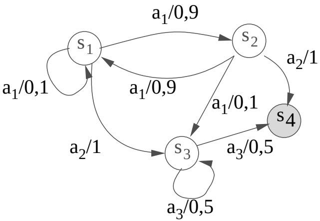


	s_i			
	s_1	s_2	s_3	s_4
$T(s_1, a1, s_i)$	0	1	0	0
$T(s_1, a2, s_i)$	0	0	1	0
$T(s_1, a3, s_i)$	1	0	0	0
$T(s_2, a1, s_i)$	1	0	0	0
$T(s_2, a2, s_i)$	0	0	0	1
$T(s_2, a3, s_i)$	0	1	0	0
$T(s_3, a1, s_i)$	0	0	1	0
$T(s_3, a2, s_i)$	0	1	0	0
$T(s_3, a3, s_i)$	0	0	0	1
$T(s_4, a1, s_i)$	0	0	0	1
$T(s_4, a2, s_i)$	0	0	0	1
$T(s_4, a3, s_i)$	0	0	0	1

$$R(s_i, a, s_j) = \begin{cases} 1 & s_j = s_4 \\ 0 & \text{en otro caso} \end{cases}$$

Ejemplo de MDP Estocástico

Las transiciones de estado y la función de refuerzo son funciones estocásticas, por lo que la misma situación puede producir distintos resultados



	s_i			
	s_1	s_2	s_3	s_4
$T(s_1, a1, s_i)$	0.1	0.9	0	0
$T(s_1, a2, s_i)$	0	0	1	0
$T(s_1, a3, s_i)$	1	0	0	0
$T(s_2, a1, s_i)$	0.9	0	0.1	0
$T(s_2, a2, s_i)$	0	0	0	1
$T(s_2, a3, s_i)$	0	1	0	0
$T(s_3, a1, s_i)$	0	0	1	0
$T(s_3, a2, s_i)$	0	0	1	0
$T(s_3, a3, s_i)$	0	0	0.5	0.5
$T(s_4, a1, s_i)$	0	0	0	1
$T(s_4, a2, s_i)$	0	0	0	1
$T(s_4, a3, s_i)$	0	0	0	1

$$R(s_i, a, s_j) = \begin{cases} 1 & s_j = s_4 \\ 0 & \text{en otro caso} \end{cases}$$

Propiedad de Markov

- **Propiedad de Markov**:
  El estado actual y el refuerzo obtenido son condicialmente independientes de la historia pasada dados el estado anterior y la acción ejecutada

$$P(s_{t+1}, r_{t+1} \mid s_t, a_t, r_t, s_{t-1}, \ldots, s_0, a_0) = P(s_{t+1}, r_{t+1} \mid s_t, a_t)$$

- **Consecuencia**: la acción a ejecutar sólo depende del estado actual

En este tema

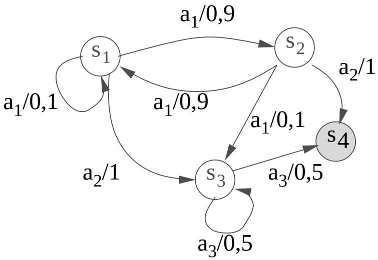
Políticas y Optimalidad

- Objetivo de planificación:
  - Encontrar una política, $\pi : S \rightarrow A$, que para cada estado $s \in S$, decida cuál es la acción, $a \in A$, que debe ser ejecutada, de forma que se maximice el refuerzo acumulado a lo largo del tiempo.
- Criterio de optimalidad de horizonte infinito descontado: dado un estado inicial arbitrario s_t, la política óptima debe maximizar

$$r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+1+k}$$

donde $0 \leq \gamma \leq 1$ es el factor de descuento

Ejemplo de Política



Estado	Acción
s_1	a_1
s_2	a_2
s_3	a_3
s_4	a_1

Función de Valor (V)

- Dada una política π y un estado inicial s, $V^{\pi}(s)$ representa el refuerzo descontado acumulado en el tiempo si se aplica la política π partiendo de s
- $V^{\pi}(s)$ se denomina función de valor-estado
- En el caso no determinista $V^{\pi}(s)$ es un valor esperado

$$V^{\pi}(s_t) = E\left[\sum_{k=0}^{\infty} \gamma^k r_{t+1+k}\right]$$

Política óptima

- La política óptima π^*, es aquella que maximiza $V^\pi(s)$ para todos los estados

$$\pi^* \equiv \underset{\pi}{\mathrm{argmax}}\, V^\pi(s), \forall(s)$$

  - Para simplificar la notación nos referiremos a la función de valor de una política óptima como $V^*(s)$

$$V^*(s) = V^{\pi^*}(s)$$

  - $V^*(s)$ es máximo refuerzo (esperado) descontado acumulado en el tiempo que el agente puede conseguir comenzando en el estado s y siguiendo la política óptima π^*
- ¡¡Queremos calcular π^*!! (resolver el MDP)

Función de valor-estado-acción (Q)

- Dada una política π, un estado inicial s y una acción a, $Q^\pi(s,a)$ representa el refuerzo (esperado) descontado acumulado en el tiempo si se aplica la acción a en el estado s, y a partir de ahí se ejecuta la política π
- Función de valor-estado-acción óptima y relación con V

$$Q^*(s,a) = \max_{\pi} Q^\pi(s,a), \forall s \in S, \forall a \in A$$

$$V^*(s) = \max_{a} Q^*(s,a), \forall s \in S$$

- Política óptima en función de Q

$$\pi^*(s) = \operatorname*{argmax}_{a \in A} Q^*(s,a), \forall s \in S$$

Ecuaciones de Bellman

- Ecuaciones recurrentes que se basan en calcular el refuerzo total óptimo maximizando sobre la elección de una primera acción y considerando un futuro óptimo
- Formalmente

$$Q^*(s,a) = \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V^*(s') \right]$$

$$V^*(s) = \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V^*(s') \right]$$

$$V^*(s) = \max_a Q^*(s,a)]$$

En este tema

Resolver el MDP: aproximaciones basadas en el modelo

- La función de transición y la función de refuerzo son conocidas
- Las ecuaciones del Bellman se resuelven por Programación Dinámica
    - Algoritmo Value Iteration
    - Algoritmo Policy Iteration

Value Iteration

Value iteration

- Inicializar: $V(s) = 0$ para todo s
- Repetir hasta convergencia
  1. Dados $V(s)$ para todo s, actualizar

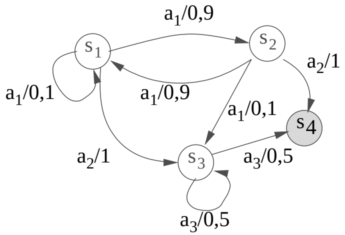
$$V(s) = \max_a \sum_{s'} T(s, a, s') \left[R(s, a, s') + \gamma V(s')\right] \ , \forall s$$

  2. Política generada en la iteración

$$\pi(s) = \operatorname{argmax}_a \sum_{s'} T(s, a, s') \left[R(s, a, s') + \gamma V(s')\right] \ , \forall s$$

Cuando el algoritmo converge (los valores se estabilizan) la política es óptima

Ejemplo



Inicialización: $V(s_i) = 0 \, \forall s_i$

Iteración 1:

$$V(s_2) = \text{máx}\{ \quad 0{,}9 \times [R(s_2, a_1, s_1) + \gamma V(s_1)] + 0{,}1 \times [R(s_2, a_1, s_3) + \gamma V(s_3)],$$
$$1 \times [R(s_2, a_2, s_4) + \gamma V(s_4)]\} = \text{máx}\{0, 1\} = 1$$
$$\pi(s_2) = a_2$$

$$V(s_1) = \text{máx}\{ \quad 0{,}1 \times [R(s_1, a_1, s_1) + \gamma V(s_1)] + 0{,}9 \times [R(s_1, a_1, s_2) + \gamma V(s_2)],$$
$$1 \times [R(s_1, a_2, s_3) + \gamma V(s_3)]\} = \text{máx}\{0, 0\} = 0$$
$\pi(s_1) = a_1$, (podría ser , a_2)

$$V(s_3) = \text{máx}\{ \quad 0{,}5 \times [R(s_3, a_3, s_3) + \gamma V(s_3)] + 0{,}5 \times [R(s_3, a_3, s_4) + \gamma V(s_4)]$$
$$\text{máx}\{0{,}5\} = 0{,}5$$
$$\pi(s_3) = a_3$$

Iteración 2:...

...

Policy Iteration

Policy Iteration

- Elegir una política π arbitraria
- Repetir hasta convergencia
  1. **Evaluación de la política**: resolver las ecuaciones $\forall s \in S$

$$V(s) = \sum_{s'} T(s, \pi(s), s') \left[R(s, \pi(s), s') + \gamma V(s')\right]$$

  2. **Mejora de la política** $\forall s \in S$

$$\pi(s) = \operatorname*{argmax}_{a} \sum_{s'} T(s, a, s') \left[R(s, a, s') + \gamma V(s')\right]$$

Cuando el algoritmo converge (la política se estabiliza) la política es óptima

En este tema

Modelo desconocido. Aprendizaje por Refuerzo

- Problema de Aprendizaje por Refuerzo (definido como un MDP):
  - Conjunto de todos los posibles estados, S,
  - Conjunto de todas las posibles acciones, A,
  - Función de transición de estados desconocida, $T : S \times A \times S \to \mathbb{R}$
  - Función de refuerzo desconocida, $R : S \times A \times S \to \mathbb{R}$
- Objetivo: aprender la política de acción $\pi : S \to A$ que maximice el refuerzo esperado acumulado en el tiempo

Aproximaciones Basadas en el Modelo

1. Se aprende el modelo (función de transición y función de refuerzo) y se resuelven las ecuaciones por Programación Dinámica (**Value Iteration**, **Policy Iteration**)
   - Técnicas que pueden ser costosas computacionalmente
   - No útil si se desean respuestas en tiempo real
   - Se debe asegurar que se aprende el entorno completamente
   - No es sensible a cambios en el entorno
2. Se aprende el modelo a la vez que la función Q (algoritmo **Dyna-Q**)

Aproximaciones libres de Modelo

- Actualización directa de la función de valor Q a partir de interaciones con el entorno
- Basados en procesos de prueba y error
- NO aprenden el MDP subyacente
- Métodos Monte Carlo y métodos de Diferencia Temporal (Q-learning, SARSA)

Métodos Monte Carlo (MC)

- Objetivo: estimar Q^*
- 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
  - Se actualiza Q al final de cada episodio

Monte Carlo con Arranque Exploratorio

Monte Carlo ES

- Inicializar, para todo $s \in S$, $a \in A$:
  - $Q(s, a) \leftarrow$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
     - $R(s, a) \leftarrow$ refuerzo descontado acumulado tras la primera ocurrencia del par (s, a)
     - Añadir $R(s, a)$ a $ganancias(s, a)$
     - $G = promedio(ganancias(s, a))$
     - $Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha G$
       (Caso determinista $\alpha = 1$)
  3. Para cada s en el episodio
     $\pi(s) \leftarrow \text{argmax}_a Q(s, a)$

Métodos de Diferencia Temporal (TD)

- Objetivo: estimar Q^*
- 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 Q
    - Se actualiza Q en cada paso del episodio
    - Estimaciones calculadas sobre estimaciones
- Algoritmos:
    - Q-Learning: off-policy
    - SARSA: on-policy

Q-Learning (Watkins, 1989).

- Se parte de una tabla $Q(s, a)$ inicial
- Idea principal: actualizar $Q(s, a)$ tras la ejecución de cada acción (transición s, a, r, s')
  - Definición de Q

$$Q(s, a) = r + \gamma V(s')$$

  - Estimar $V(s')$ utilizando la tabla Q actual

$$Q(s, a) = r + \gamma \max_{a'} Q(s', a')$$

Funciones de Actualización de Q

- Supongamos la transición s, a, r, s'
- Función de actualización determinista

$$Q(s,a) \leftarrow r + \gamma \max_{a'} Q(s', a')$$

- Función de actualización no determinista (caso general)

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

$\alpha \in [0, 1]$ es la tasa de aprendizaje

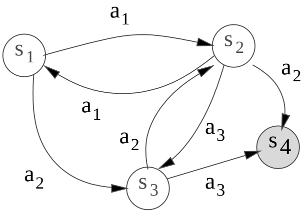
Q-Learning (Watkins, 1989)

Q-Learning (γ, α).

Inicializar $Q(s, a)$, $\forall s \in \mathcal{S}, a \in \mathcal{A}$ (lo habitual es inicializar a 0)
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 $Q(s, a)$

Ejemplo

- Dado el siguiente MDP determinista



$$R(s_i, a, s_j) = \begin{cases} 1 & s_j = s_4 \\ 0 & \text{en otro caso} \end{cases}$$

- 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

- En este ejemplo asumimos $\gamma = 0{,}5$
- 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:
  - $Q(s_1, a_1) = R(s_1, a_1) + \gamma \max_a Q(s_2, a) = 0 + \gamma 0 = 0$
  - $Q(s_2, a_3) = R(s_2, a_3) + \gamma \max_a Q(s_3, a) = 0 + \gamma 0 = 0$
  - $Q(s_3, a_3) = R(s_3, a_3) + \gamma \max_a Q(s_4, a) = 1 + \gamma 0 = 1$
- 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:
  - $Q(s_1, a_2) = R(s_1, a_2) + \gamma \text{ máx}_a Q(s_3, a) =$
    $0 + \gamma \text{ máx}(0, 0, 1) = \gamma = 0{,}5$
  - $Q(s_3, a_2) = R(s_3, a_2) + \gamma \text{ máx}_a Q(s_2, a) = 0 + \gamma 0 = 0$
  - $Q(s_2, a_2) = R(s_2, a_2) + \gamma \text{ máx}_a Q(s_4, a) = 1 + \gamma 0 = 1$
- 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:
  - $\pi^*(s_3) = \text{argmax}_a\, Q(s_3, a) = a_3$
  - $\pi^*(s_2) = a_2$
  - $\pi^*(s_1) = a_1$
- Otra política óptima: igual que la anterior pero con $\pi^*(s_1) = a_2$

Exploración vs. Explotación

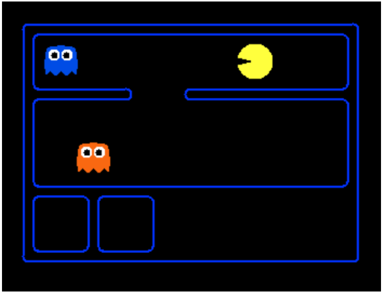
- Estrategias de selección de acciones
  - ϵ-greedy
    Ejecuta una acción aleatoria con probabilidad ϵ y la acción $\text{argmax}_a Q(s,a)$ con probabilidad $1-\epsilon$
  - Softmax
    $$P(a_i \mid s) = \frac{e^{Q(s,a_i)/\tau}}{\sum_{a_j \in \mathcal{A}} e^{Q(s,a_j)/\tau}}$$
- Inicialización de la función Q
- Sesgar la selección de acciones con conocimiento del dominio adicional

Aplicaciones

- Planificación: tiempo real, entornos estocásticos
- Control de dispositivos de salud
- Control de robots: navegación, manipulación
- Control de procesos de fabricación
- Control de temperatura
- Predicción de series: reconocimiento de voz, predicción de mercado
- Juegos: AlphaGo, TD-Gammon
- Logística
- Problemas canónicos: navegación en un grid, balance de un sistema *cart-pole*

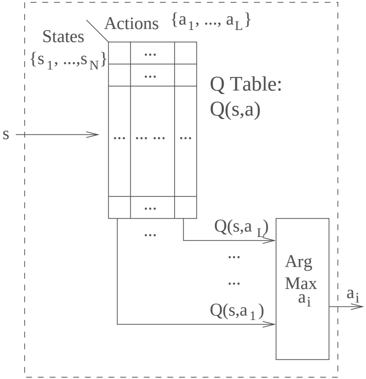
En este tema

Pac-Man



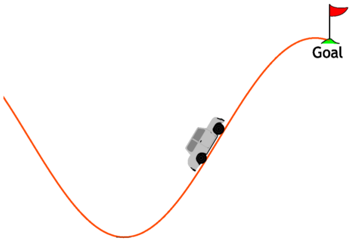
- Observaciones: coordenadas discretas x e y de Pac-Man y fantasmas
- Acciones: movimientos de tamaño 1 en las 4 direcciones
- Objetivo: comer fantasmas

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

Mountain-Cart

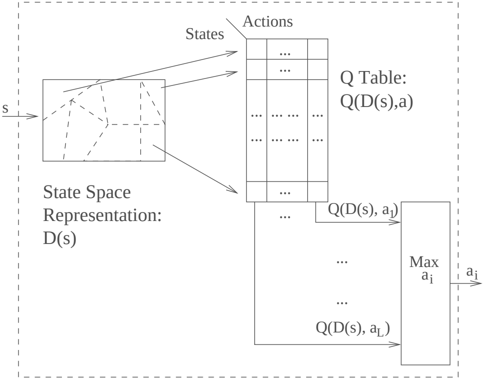


- Observaciones: valores continuos de posición x y velocidad v del coche
- Acciones: aplicación de fuerza hacia la derecha, izquierda, o nula
- Objetivo: alcanzar la bandera con velocidad 0

En este tema

En este tema

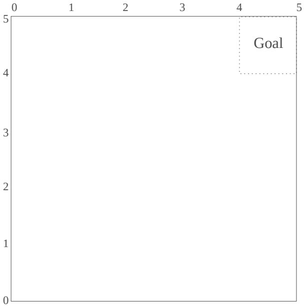
Discretización del Espacio de Estados

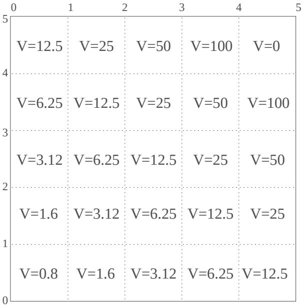


- Problema:
  - Discretizaciones erróneas pueden romper fácilmente la propiedad de Markov
  - Cuántas regiones necesitamos para discretizar el espacio de estados?

Ejemplo de Discretización Uniforme

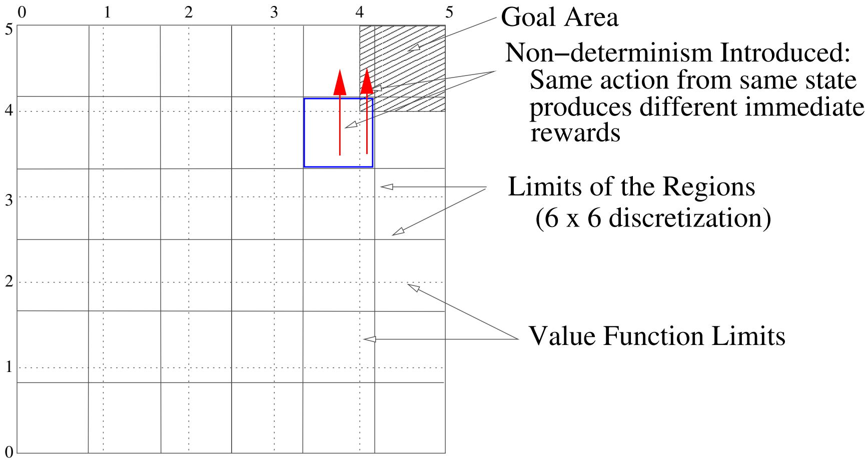
- Dominio de navegación de un robot:
  - Espacio de estados continuo de tamaño 5×5
  - Acciones: Norte, Sur, Este, Oeste, de tamaño 1





  - Discretización óptima de tamaño 5×5

Pérdida de la Propiedad de Markov



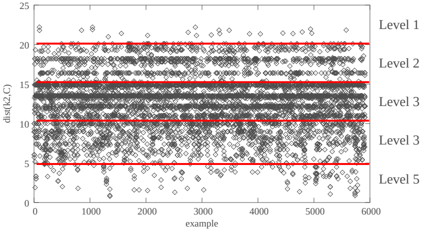
Keepaway (Stone, Sutton and Kuhlmann, 05)

Ejemplo: la Tarea Keepaway

- Espacio de estados: 19 atributos continuos (Los keepers y los takers se ordenan tomando en cuenta su distancia al jugador)
  - $\text{dist}(k_1, C), \ldots, \text{dist}(k_4, C)$
  - $\text{dist}(t_1, C), \ldots, \text{dist}(t_3, C)$
  - $\text{dist}(k_1, t_1), \ldots, \text{dist}(k_1, t_3)$
  - $\text{Min}(\text{dist}(k_2, t_1), \text{dist}(k_2, t_2), \text{dist}(k_2, t_3))$
  - etc. . .
- Espacio de acciones 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

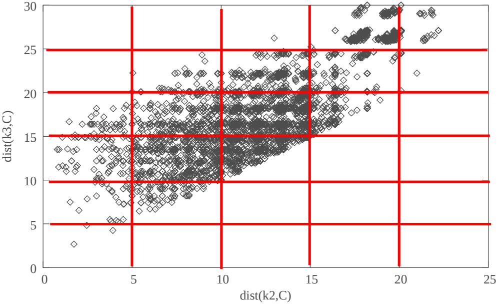
Discretización uniforme del espacio de estados

- 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

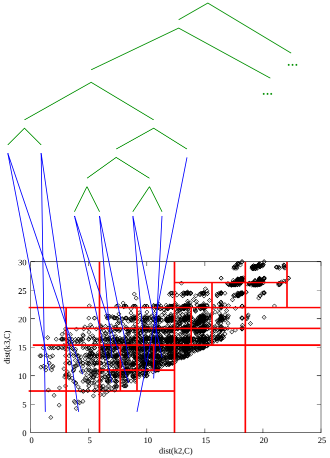
- Proyección del espacio de estados sobre los atributos 2 y 3:



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
  - ...



Clustering para Q-Learning

- Discretización no uniforme del espacio de estados (k-means)
- Los clusters obtenidos se consideran la nueva representación del espacio de estados
- Se aplica Q-Learning sobre la nueva representación

Clustering para Q-learning