Aprendizaje Reforzado 2

Lección 25

Dr. Pablo Alvarado Moya

CE5506 Introducción al reconocimiento de patrones
Área de Ingeniería en Computadores
Tecnológico de Costa Rica

II Semestre, 2019

Contenido

Repaso (1)

- Vimos los **MDP** (procesos de decisión de Markov) como formalismo para el aprendizaje reforzado.
- Un MDP es una tupla $\langle \mathcal{S}, \mathcal{A}, \{P_{sa}\}, \gamma, R \rangle$ con
  - Estados $\mathcal{S}$
  - Acciones $\mathcal{A}$
  - Probabilidades de transición P_{sa}
  - Factor de degradación γ
  - Función de recompensa R.
- Una **política** π es una función $\pi : \mathcal{S} \to \mathcal{A}$
- La **función de valor** para una política π es $V^\pi(s) : \mathcal{S} \to \mathbb{R}$

$$V^\pi(s) = \mathrm{E}\left[R(s_{(0)}) + \gamma R(s_{(1)}) + \gamma^2 R(s_{(2)}) + \cdots | s_{(0)} = s, \pi\right]$$

Repaso (2)

- La estrategia del aprendizaje es encontrar primero

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

  y luego con ello encontrar la política π^*

$$\pi^*(s) = \arg\max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} P_{sa}(s')V^*(s')$$

- El proceso de búsqueda de V^* usa las ecuaciones de Bellman para la función de valor óptima:

$$V^*(s) = R(s) + \max_{a \in \mathcal{A}} \gamma \sum_{s' \in \mathcal{S}} P_{sa}(s')V^*(s')$$

Repaso (3)

- En el algoritmo de **iteración de valor**, inicializamos $V(s) = 0$ o con un valor previo, y luego repetimos hasta convergencia

$$V(s) \leftarrow R(s) + \max_{a \in \mathcal{A}} \gamma \sum_{s'} P_{sa}(s') V(s')$$

  que al final será equivalente a $V^*(s)$

- En el algoritmo de **iteración de política** inicializamos π aleatoriamente, y luego repetimos los siguiente dos pasos hasta convergencia:
  1. $V \leftarrow V^\pi$
  2. Actualice $\pi(s) = \arg\max_{a \in \mathcal{A}} \sum_{s'} P_{sa}(s') V(s')$
- Lo que haremos ahora es extender los dos algoritmos

Estados contínuos

- Hasta ahora hemos visto MDP con un número finito de estados.
- Ahora vamos a usar MDP con un número infinito de estados.
- Ejemplos:
    - Podemos representar el estado de un automóvil como $(x, y, \theta, \dot{x}, \dot{y}, \dot{\theta})$
        - (x, y), $(\dot{x}, \dot{y})$ posición y velocidad del automóvil.
        - θ, $\dot{\theta}$ orientación y velocidad angular.
        - $\mathcal{S} = \mathbb{R}^6$ es en este caso el **espacio de estados** y es continuo.
    - El estado de un péndulo invertido sería $(x, \theta, \dot{x}, \dot{\theta})$ y $\mathcal{S} = \mathbb{R}^4$ el espacio de estados.
    - El estado de un helicóptero sería $(x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi})$ y $\mathcal{S} = \mathbb{R}^9$
- Obviamente tenemos un número infinito de estados.

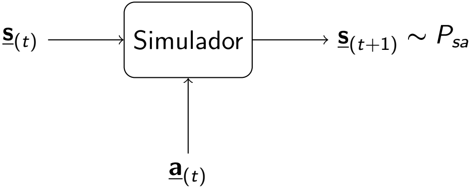
Discretización

- Una primera estrategia para usar los MDP de estados continuos es la discretización.
- El espacio $\mathcal{S}$ se discretiza, y se asume que las funciones de valor y las políticas en cada celda $\underline{\bar{\mathbf{s}}}$ son constantes.
- Se utilizan entonces los métodos ya vistos para MDP de estados discretos para encontrar $V^*(\underline{\bar{\mathbf{s}}})$, $\pi^*(\underline{\bar{\mathbf{s}}})$.
- Problemas:
  - Se introducen errores de cuantificación
  - Hay que lidiar con la maldición de la dimensión
- Como regla empírica, la discretización funciona en problemas 1D y 2D. Con muchas suerte y destrezas empíricas podría hacerse funcionar con 6D como máximo.

Aproximación de la función de valor

- Vamos a presentar métodos que calculan $V^*(\underline{\mathbf{s}})$ directamente, sin necesidad de recurrir a discretización
- En el siguiente problema, asumiremos $\underline{\mathbf{s}} \in \mathcal{S}$ continuo, pero $\underline{\mathbf{a}} \in \mathcal{A}$ discreto.
- En la praxis, las dimensiones de $\mathcal{A}$ son (mucho) menores que las dimensiones de $\mathcal{S}$.
- Si $\underline{\mathbf{a}}$ fuese continuo, usualmente puede discretizarse sin mayor consecuencia.
- Supondremos que tenemos un **modelo** o **simulador** del MDP

Simulador



- El **simulador** o **modelo** del MDP es una **caja negra** que toma como entrada cualquier valor (continuo) $\underline{\mathbf{s}}_{(t)}$, una acción $\underline{\mathbf{a}}_t$ y produce el siguiente estado $\underline{\mathbf{s}}_{(t+1)}$, de acuerdo a las probabilidades de transición $P_{\underline{\mathbf{s}}_{(t)}\underline{\mathbf{a}}_{(t)}}$.

Modelo determinístico

- El modelo puede ser el resultado del análisis de sistema.
- Para el caso del péndulo invertido tenemos por ejemplo:

$$\underline{\mathbf{s}} = \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \qquad \dot{\underline{\mathbf{s}}} = \begin{bmatrix} \dot{x} \\ \alpha - L\beta\cos(\theta)/M \\ \dot{\theta} \\ \beta \end{bmatrix} \qquad \underline{\mathbf{s}}_{(t+1)} = \underline{\mathbf{s}}_{(t)} + \dot{\underline{\mathbf{s}}}_{(t)}\Delta t$$

  con α y β magnitudes físicas dependientes del estado.
- Este modelo es **determinístico** (no hay incertidumbres estocásticas).
- Otra posibilidad para obtener el modelo es **aprender** uno.

Aprender un modelo (1)

- Para aprender el modelo necesitamos plantear un problema de aprendizaje supervisado.
- Iniciamos en un estado $\underline{\mathbf{s}}_{(0)}$ y le aplicamos una acción $\underline{\mathbf{a}}_{(0)}$ para llegar a un nuevo estado $\underline{\mathbf{s}}_{(1)}$ y así sucesivamente, de acuerdo a una **política** dada.
- Obtenemos múltiples **trayectorias** ejecutando distintas políticas

$$\underline{\mathbf{s}}_{(0)}^{(1)} \xrightarrow{\underline{\mathbf{a}}_{(0)}^{(1)}} \underline{\mathbf{s}}_{(1)}^{(1)} \xrightarrow{\underline{\mathbf{a}}_{(1)}^{(1)}} \underline{\mathbf{s}}_{(2)}^{(1)} \xrightarrow{\underline{\mathbf{a}}_{(2)}^{(1)}} \underline{\mathbf{s}}_{(3)}^{(1)} \cdots$$

$$\underline{\mathbf{s}}_{(0)}^{(2)} \xrightarrow{\underline{\mathbf{a}}_{(0)}^{(2)}} \underline{\mathbf{s}}_{(1)}^{(2)} \xrightarrow{\underline{\mathbf{a}}_{(1)}^{(2)}} \underline{\mathbf{s}}_{(2)}^{(2)} \xrightarrow{\underline{\mathbf{a}}_{(2)}^{(2)}} \underline{\mathbf{s}}_{(3)}^{(2)} \cdots$$

$$\vdots$$

Aprender un modelo (2)

- Luego usamos un algoritmo de aprendizaje para estimar $\underline{\mathbf{s}}_{(t+1)}$ en función de $\underline{\mathbf{s}}_{(t)}$ y $\underline{\mathbf{a}}_{(t)}$.
  Por ejemplo: considerando que $\underline{\mathbf{s}}$ es un vector en 4D, usamos cuatro regresores lineales que utilizan $\underline{\mathbf{s}}_{(t)}$ y $\underline{\mathbf{a}}_{(t)}$ como entradas, y predicen las salidas:

$$\underline{\mathbf{s}}_{(t+1)} = \mathbf{A}\underline{\mathbf{s}}_{(t)} + \mathbf{B}\underline{\mathbf{a}}_{(t)}$$

  Buscamos entonces las matrices $\mathbf{A}$ y $\mathbf{B}$ tales que

$$\min_{\mathbf{A},\mathbf{B}} \sum_{i=1}^{m} \sum_{t=0}^{T-1} \left\| \underline{\mathbf{s}}_{(t+1)}^{(i)} - \left(\mathbf{A}\underline{\mathbf{s}}_{(t)}^{(i)} + \mathbf{B}\underline{\mathbf{s}}_{(t)}^{(i)} \right) \right\|_2^2$$

Aprender un modelo (3)

- Alternativamente pueden usarse modelos no lineales:
  1. Una posibilidad es usar

     $$\underline{\mathbf{s}}_{(t+1)} = \mathbf{A}\phi_s(\underline{\mathbf{s}}_{(t)}) + \mathbf{B}\phi_a(\underline{\mathbf{a}}_{(t)})$$

     con ϕ_s y ϕ_a mapeos no lineales.
  2. Otros autores usan la regresión ponderada localmente.
- Filtros de Kalman son una alternativa del procesamiento de señales adaptativo ideal para este tipo de aplicaciones.

Modelos estocástico y determinístico

- Luego de entrenar el simulador, tendríamos el modelo determinístico:

$$\underline{\mathbf{s}}_{(t+1)} = \mathbf{A}\underline{\mathbf{s}}_{(t)} + \mathbf{B}\underline{\mathbf{a}}_{(t)}$$

  o el modelo estocástico

$$\underline{\mathbf{s}}_{(t+1)} = \mathbf{A}\underline{\mathbf{s}}_{(t)} + \mathbf{B}\underline{\mathbf{a}}_{(t)} + \underline{\varepsilon}_{(t)}$$

  con $\underline{\varepsilon}_{(t)} \sim \mathcal{N}(0,\mathbf{\Sigma})$
- El filtro de Kalman estima el modelo estocástico directamente.

Iteración de valor ajustada (1)

Fitted value iteration

- Usaremos la **iteración de valor ajustada** para aproximar la función de valor V^* de un MDP de estado continuo.
- Considerando que estamos ahora en un espacio de estado continuo, la iteración de valor se plantea como la actualización:

$$\begin{aligned} V(\underline{\mathbf{s}}) :=& R(\underline{\mathbf{s}}) + \gamma \max_a \int_{\underline{\mathbf{s}}'} P_{sa}(\underline{\mathbf{s}}') V(\underline{\mathbf{s}}')\, d\underline{\mathbf{s}}' \\ =& R(\underline{\mathbf{s}}) + \gamma \max_a \mathrm{E}_{\underline{\mathbf{s}}' \sim P_{sa}} [V(\underline{\mathbf{s}}')] \end{aligned}$$

- Este paso se va a aproximar con una muestra finita de estados $\underline{\mathbf{s}}_{(i)}^{(1)} \ldots \underline{\mathbf{s}}_{(i)}^{(m)}$.

Iteración de valor ajustada (2)
Fitted value iteration

- Con aprendizaje supervisado, aproximamos la función de valor como una combinación lineal o no lineal de los estados

$$V(\underline{\mathbf{s}}) = \underline{\boldsymbol{\theta}}^T \phi(\underline{\mathbf{s}})$$

donde ϕ es algún mapeo de características de los estados, como por ejemplo:

$$\phi(\underline{\mathbf{s}}) = \underline{\mathbf{s}} \quad \text{ó} \quad \phi(\underline{\mathbf{s}}) = \begin{bmatrix} x & x^2 & \dot{x} & \dot{x}^2 & \dot{x}x & \theta & \dot{\theta} & \dots \end{bmatrix}^T$$

Algoritmo de iteración de valor ajustada

Elija aleatoriamente m estados $\underline{\mathbf{s}}^{(1)}, \underline{\mathbf{s}}^{(2)}, \ldots, \underline{\mathbf{s}}^{(m)} \in \mathcal{S}$
Inicialice $\underline{\boldsymbol{\theta}} := \underline{\mathbf{0}}$
repeat
  for $i = 1, \ldots, m$ **do**
    for cada acción $\underline{\mathbf{a}} \in \mathcal{A}$ **do**
      Muestree $\underline{\mathbf{s}}'_{(1)}, \ldots, \underline{\mathbf{s}}'_{(k)} \sim P_{s^{(i)}a}$ *# Usando modelo*
      Defina $q(\underline{\mathbf{a}}) := \frac{1}{k} \sum_{j=1}^{k} \left[R(\underline{\mathbf{s}}^{(i)}) + \gamma V(\underline{\mathbf{s}}'_{(j)}) \right]$
      # Estimado de $R(\underline{\mathbf{s}}^{(i)}) + \gamma \, \mathrm{E}_{\underline{\mathbf{s}}' \sim P_{s^{(i)}a}}[V(\underline{\mathbf{s}}')]$
    end for
    $y^{(i)} := \max_{\underline{\mathbf{a}}} q(\underline{\mathbf{a}})$ *# Est.* $R(\underline{\mathbf{s}}^{(i)}) + \gamma \max_{\underline{\mathbf{a}}} \mathrm{E}_{\underline{\mathbf{s}}' \sim P_{s^{(i)}a}}[V(\underline{\mathbf{s}}')]$
  end for
  # Ajuste regresión lineal para lograr $V(\underline{\mathbf{s}}^{(i)}) \approx y^{(i)}$
  $\underline{\boldsymbol{\theta}} := \arg\min_{\underline{\boldsymbol{\theta}}} \frac{1}{2} \sum_{i=1}^{m} \left(\underline{\boldsymbol{\theta}}^T \phi(\underline{\mathbf{s}}^{(i)}) - y^{(i)} \right)^2$
until (convergencia)

Recompensa

- En el algoritmo anterior eligiríamos la recompensa de acuerdo a la aplicación.
- En el péndulo invertido, por ejemplo:

$$R(\underline{\mathbf{s}}) = \begin{cases} -1 & \text{si péndulo se cae} \\ +1 & \text{si péndulo está equilibrado} \\ 0 & \text{en el resto} \end{cases}$$

- Podemos agregar más valores para condiciones que quisiéramos evitar o recompensar.

Algunos comentarios

- Si tuviéramos un modelo determinístico, ¿cómo debería cambiar el algoritmo?

Algunos comentarios

- Si tuviéramos un modelo determinístico, ¿cómo debería cambiar el algoritmo?
- En el paso 6., siempre obtendríamos el mismo valor.
- Por tanto, $k = 1$ en el paso 7

Algunos comentarios

- Si tuviéramos un modelo determinístico, ¿cómo debería cambiar el algoritmo?
- En el paso 6., siempre obtendríamos el mismo valor.
- Por tanto, $k = 1$ en el paso 7
- En general, encontrar mapeo ϕ adecuado para la regresión es difícil, y requiere buena comprensión del problema y del proceso.

Algunos comentarios

- Si tuviéramos un modelo determinístico, ¿cómo debería cambiar el algoritmo?
- En el paso 6., siempre obtendríamos el mismo valor.
- Por tanto, $k = 1$ en el paso 7
- En general, encontrar mapeo ϕ adecuado para la regresión es difícil, y requiere buena comprensión del problema y del proceso.
- En el caso de estados discretos procuramos encontrar $V \approx V^*(s)$ y a partir de él derivar la política π^*.
- Hasta ahora hemos procurado una manera de encontrar $V(\underline{\mathbf{s}})$.
- Queremos ahora un método para encontrar la política.

Política (1)

- La política óptima teóricamente se puede calcular con

$$\pi^*(\underline{\mathbf{s}}) = \arg\max_{\underline{\mathbf{a}}} \mathrm{E}_{\underline{\mathbf{s}}' \sim P_{\underline{\mathbf{sa}}}}[V^*(\underline{\mathbf{s}}')]$$

- Puesto que estado $\underline{\mathbf{s}}$ es continuo, no es posible calcular lo anterior de forma expresa para todo $\underline{\mathbf{s}}$, como lo hicimos en caso discreto.
- Dado un estado concreto, por ejemplo $\underline{\mathbf{s}} = [x,\theta,\dot{x},\dot{\theta}]^T$, solo entonces intentamos calcular la acción $\underline{\mathbf{a}} \in \mathcal{A}$ tal que

$$\underline{\mathbf{a}} = \arg\max_{\underline{\mathbf{a}}} \mathrm{E}_{\underline{\mathbf{s}}' \sim P_{\underline{\mathbf{sa}}}}[V(\underline{\mathbf{s}}')]$$

- Para calcular esto se usa algo parecido al ciclo interno de la iteración de valor ajustada: para cada acción muestreamos $\underline{\mathbf{s}}'_{(1)}, \ldots, \underline{\mathbf{s}}'_{(k)} \sim P_{\underline{\mathbf{sa}}}$ para aproximar la esperanza.

Política (2)

- Si el modelo/simulador es determinista, entonces lo podemos aproximar con una función f tal que:

$$\underline{\mathbf{s}}_{(t+1)} = f(\underline{\mathbf{s}}_{(t)}, \underline{\mathbf{a}}_{(t)})$$

- En este caso determinista, de nuevo usaríamos $k = 1$. Se simplifica entonces la acción a tomar como

$$\underline{\mathbf{a}}_{(t)} = \arg\max_{\underline{\mathbf{a}}'} V(f(\underline{\mathbf{s}}_{(t)}, \underline{\mathbf{a}}'))$$

Simulador estocástico

- Si el simulador se puede escribir como

$$\underline{\mathbf{s}}_{(t+1)} = f(\underline{\mathbf{s}}_{(t)}, \underline{\mathbf{a}}_{(t)}) + \underline{\varepsilon}_{(t)}$$

con $f(\underline{\mathbf{s}}_{(t)}, \underline{\mathbf{a}}_{(t)})$ una función determinista y $\underline{\varepsilon}_{(t)}$ ruido gaussiano de media nula, entonces la acción a elegir es

$$\begin{aligned}\underline{\mathbf{a}}_{(t)} &= \arg\max_{\underline{\mathbf{a}}'} V(f(\underline{\mathbf{s}}_{(t)}, \underline{\mathbf{a}}')) \\ &= \arg\max_{\underline{\mathbf{a}}'} \left[\underline{\boldsymbol{\theta}}^T \phi\left(\underline{\mathbf{s}}_{(t)}\right)\right]\end{aligned}$$

donde usamos que si $\underline{\varepsilon}_{(t)}$ es suficientemente pequeño entonces

$$\mathrm{E}_{\underline{\mathbf{s}}'}[V(\underline{\mathbf{s}}')] \approx V(\mathrm{E}_{\underline{\mathbf{s}}' \sim P_{\underline{\mathbf{sa}}}}[\underline{\mathbf{s}}']) = V(f(\underline{\mathbf{s}}_{(t)}, \underline{\mathbf{a}}_{(t)}))$$

Limitación

- En problemas que se apartan ya sea del modelo determinista con con ruido muy fuerte, requerimos muestrar $k|\mathcal{A}|$ estados para aproximar la esperanza de V correctamente.
- Esto puede representar un alto costo computacional.

Resumen

1. Repaso
2. MDP de estados continuos
   - Discretización
   - Aproximación de la función de valor

Este documento ha sido elaborado con software libre incluyendo LaTeX, Beamer, GNUPlot, GNU/Octave, XFig, Inkscape, GNU-Make y Subversion en GNU/Linux