UNIVERSIDAD DEL VALLE DE GUATEMALA
CC3104 - Aprendizaje por Refuerzo
Sección 21
Ing. Javier Josué Fong Guzmán

Programación Dinámica

- Laboratorio 4 -

José Pablo Orellana – 21970
Diego Alberto Leiva - 21572

Guatemala, 17 de julio del 2025

Introducción

En el presente laboratorio tuvimos como objetivo principal el implementar un algoritmo de iteracion de políticas, que es una técnica de programación dinámica que se usa en aprendizaje por refuerzo para poder encontrar políticas óptimas en un entorno ya definido. Se trabajó con un entorno tipo GridWorld y un agente que debía completar los métodos policylterate y policyimprove, el algoritmo se probó en el arvhivo llamado test, validando la convergencia de la política y los valores de estado según lo descrito en *Reinforcement Learning: An Introduction* de Sutton y Barto. (Sutton & Barto, 2018).

Descripción feneral de la implementacion de policyIterate y policyImprove

En el archivo agent.py, se implementó el método policyIterate siguiendo los pasos:

- **Evaluación de la política actual**: se utilizó un ciclo que actualiza los valores de estado (V) hasta que el cambio máximo entre iteraciones sea menor a un umbral de convergencia. Esto se implementa mediante una iteración sobre todos los estados no terminales, calculando el valor esperado de cada acción de acuerdo con la política actual y el modelo del entorno (env.P definido en gridworld.py).
- Llamada a policyImprove después de estabilizar los valores para mejorar la política.

En el mismo archivo, policyImprove actualiza la política (policy) seleccionando para cada estado la acción que maximiza el valor esperado, usando la ecuación de Bellman. El cambio de política se mide para determinar si se ha alcanzado la política óptima.

Resultados obtenidos al ejecutar el agente

En el archivo test.ipynb, al ejecutar el agente con el entorno GridWorld, la política converge hacia un comportamiento óptimo en el que el agente se desplaza hacia el objetivo de la manera más eficiente posible, evitando estados de penalización.

Los valores de estado (V(s)) muestran un incremento gradual conforme se aproximan a la meta, lo que refleja correctamente el valor esperado acumulado. La política final se estabilizó en pocas iteraciones (dependiendo del parámetro de convergencia definido en policyIterate).

Breve análisis sobre la convergencia de la política y los valores

La convergencia se alcanzó cuando los valores de todos los estados cambiaron por debajo del umbral definido (theta), lo que indica que se llegó a una **política estable**. En policyIterate, la

condición de estabilidad de política se cumple cuando policyImprove no realiza cambios adicionales en la política actual.

Este comportamiento es consistente con la teoría: primero, la evaluación de política converge más rápido cuando el factor de descuento (gamma) es menor que 1; segundo, la mejora de política garantiza que no se retroceda a una política peor, asegurando convergencia en un número finito de iteraciones.

Observaciones o dificultades encontradas durante la implementación

Durante la implementación se identificaron los siguientes puntos:

- Fue necesario entender la estructura de env.P en gridworld.py, que define las transiciones de estado y recompensas, para calcular correctamente los valores esperados en ambos métodos.
- La convergencia depende fuertemente del valor de theta y gamma; valores demasiado pequeños aumentaban las iteraciones necesarias.
- Se realizaron pequeñas modificaciones en test.ipynb para visualizar mejor la política y los valores, lo que facilitó la depuración.
- Al principio, un error común fue no diferenciar correctamente entre evaluación de política y mejora de política, lo que llevaba a bucles infinitos; se corrigió asegurando que policyImprove se ejecutara solo después de estabilizar valores.

Bibliografía

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). The MIT Press. https://goo.su/KUVJH