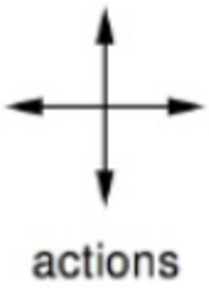
인공지능 Lab. Project #2

● Dynamic Programming

본 프로젝트에서는 Dynamic Programming(DP)을 이용하여 environment의 model을 푸는 과제이다. DP는 큰 Programming을 여러 process로 나누어 해결한다는 의미이다. 큰 순차적인 행동 결정을 작은 process로 나누어 제시된 grid world에서 Policy iteration과 Value iteration을 policy를 최적화 시켜보는 것이 과제입니다. 7x7의 시작점과 끝점을 가진 gird-world입니다. Action은 상하좌우 4가지입니다. 제일 바깥쪽 state에서 gird-world 밖으로 나가는 action을 취할 경우, 제자리로 돌아옵니다. 중간중간에 함정이 존재합니다.



START	-1	-100	-1	-1	-1	-1
-1	-1	-100	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-100	-100	-1
-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1
-1	-1	-100	-100	-1	-1	**END**

그림 1. Grid world example

Reward를 최대로 받아야하는 agent는 최단거리로 종료점까지 가는 경로를 학습할 것입니다. 처음 agent가 택하는 policy는 상하좌우 모두 같은 확률로 uniform random policy 입니다. 그럴 경우 그림 2처럼 처음에 계산하게 될 것입니다.

START	-1	-100	-1	-1	-1	-1
-1	-1	-100	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-100	-100	-1
-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1
-1	-1	-100	-100	-1	-1	**END**

Highlight point (1,2) state

Up $V_2(s) = 0.25 \times (-1 + -1)$
Down $V_2(s) = 0.25 \times (-1 + -1)$
Left $V_2(s) = 0.25 \times (-1 + 0)$
Right $V_2(s) = 0.25 \times (-1 + -1)$

$\therefore V_2(s) = 3 \times 0.25 \times (-2) + 0.25 \times (-1) = \mathbf{-1.75}$

그림 2. 처음 agent가 택하는 policy evaluation 계산 example

□ **Program implementation**

1. Policy evaluation _ random Policy

Policy evaluation을 구현해주시면 됩니다. Policy evaluation은 현재 주어진 Policy에 대한 value function을 구하는 것이고, one step backup으로 구현합니다. 아래 그림에서 k는 iteration number를 의미하고 one step씩 각 state의 value function을 update하는 과정입니다.

4x4 grid world example로 예시를 들면 다음 그림 3과 같습니다.

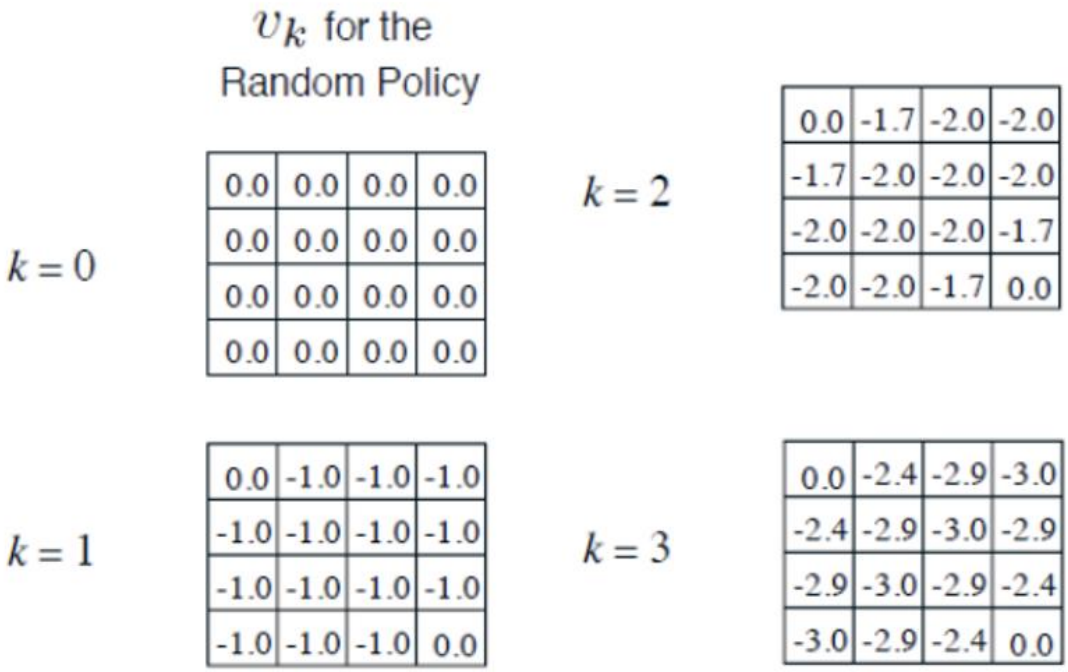


그림 3. Value function의 update과정

한 step씩 다음 식을 통해서 value function을 update합니다.

$$v_{k+1}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_k(s') \right)$$

$3 \times 0.25 \times (-1 + (-1)) + 0.25(-1+0)$
$= -6 \times 0.25 - 0.25 = -1.75$
⋮

이런 식을 무한대까지 계산하게 되면 현재 random policy에 대한 true value function을 구할 수 있습니다. 이번 과제에서는 위에서 제안한 7x7 grid world에서 다음과 같이 policy evaluation을 구현하는 것이 목표입니다. Policy evaluation을 구현하여 주시고 k의 값을 0,1,2,3과 같이 초반의 변하는 부분과 충분히 k의 값을 늘려 수렴하게 된 부분을 모두 캡처하여 보고서에 작성하여 주십시오.

2. Policy Improvement

Iteration을 반복하며 true value function을 찾았다면, 이 policy를 따르는 것이 좋을지 안 좋을지를 판단하고 Policy를 update해야합니다. 이를 통해서 현재 policy보다 더 나은 policy를 찾아가면 optimal policy에 가까워지게 됩니다. 이 과정을 Policy Improvement라고 합니다. 널리 알려진 Greedy policy improvement로 구현하겠습니다. max값만을 선택하는 방법으로 policy evaluation에서 모든 state의 value값을 구해 놓았다면, 이제는 policy에 따른 action을 취해야합니다. 즉 현재 state에서 가장 높은 곳으로 이동하는 action을 취하겠다는 의미입니다.

위의 예제에서 이어서 설명하면, Highlight된 state에서 max값을 선택하여 Left 방향이 구해진 것을 확인할 수 있습니다.

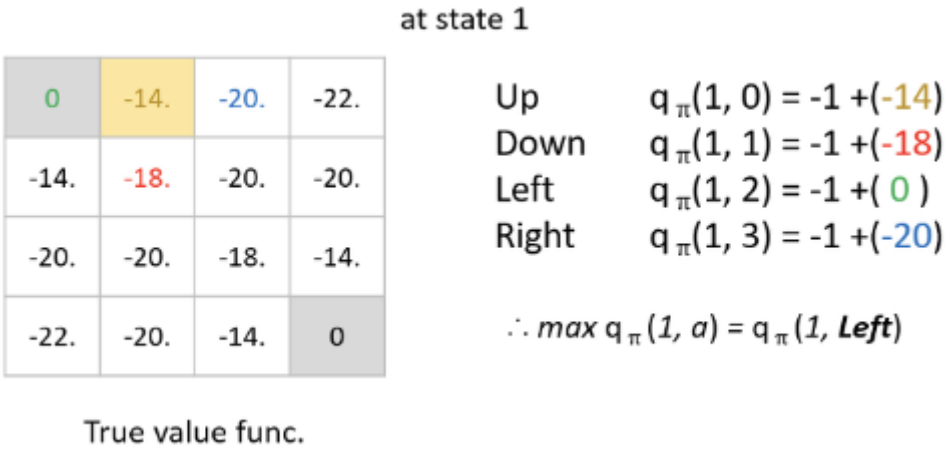


그림 4. True value function

이런 policy evaluation과 policy improvement를 반복하여 optimal policy를 찾게 됩니다. 이것을 Policy Iteration이라고 부르게 됩니다. 위와 같이 제안된 7x7 grid world에서 기존의 계산한 policy evaluation의 true value function을 이용하여 policy과 업데이트되는 과정과 각각 state에서의 action을 매트릭스로 만들어 보고서에 작성하여 주십시오.

3. Value Iteration

Value Iteration은 Bellman Optimality Eqn.을 이용하여 계산합니다. Evaluation 과정에서 value function의 action을 취할 확률을 곱해서 합치는 것 대신에 max값을 취해서 greedy하게 value function을 구해서 improve해버리자는 것이 Value Iteration의 아이디어입니다. 4x4 grid world에서 계산하는 예시를 들면 다음 그림 5와 같습니다.

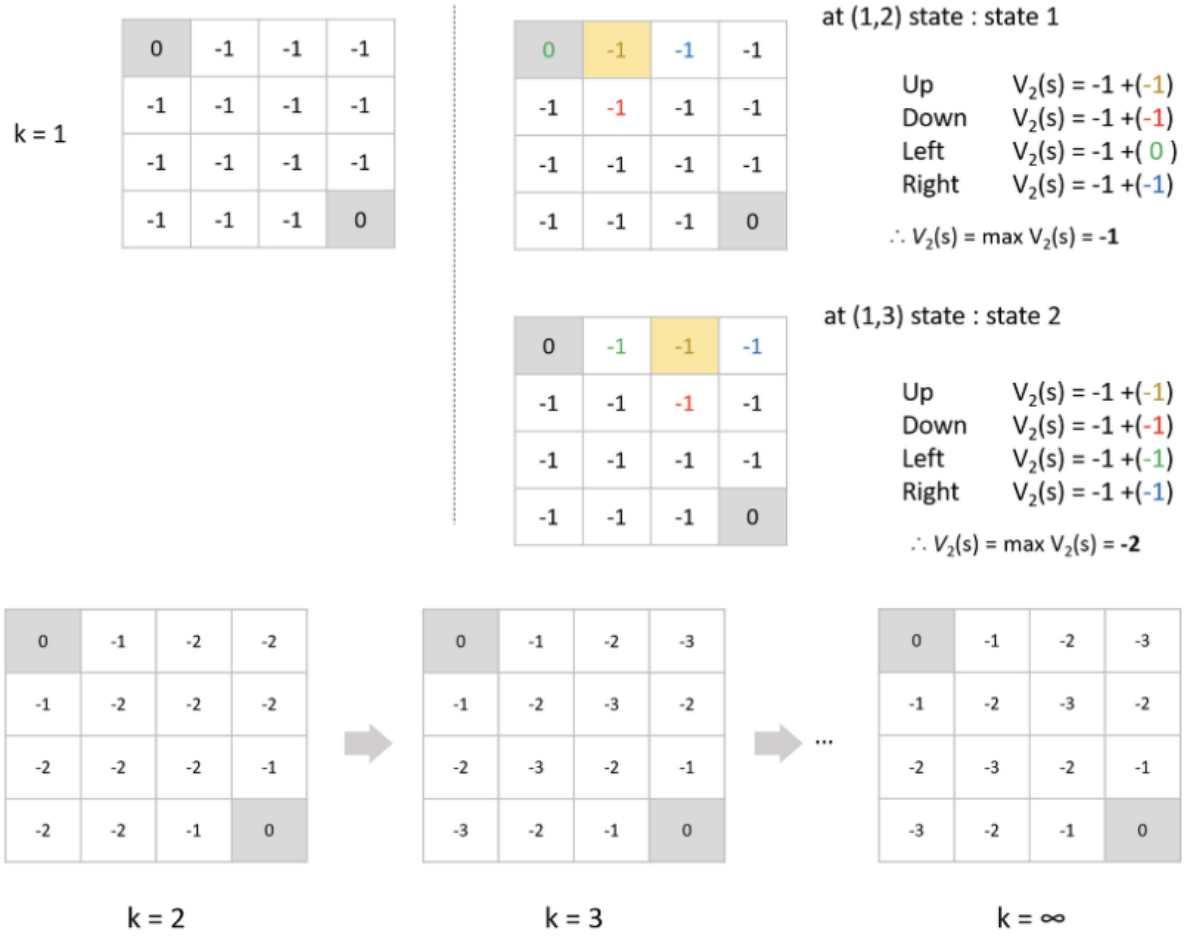


그림 5. Value Iteration

이와 같은 Value Iteration 방법을 과제에서 제안된 7x7 grid world로 구현하고, k의 값을 0,1,2,3과 같이 초반의 변하는 부분과 충분히 k의 값을 늘려 수렴하게 된 부분을 모두 캡처하여 보고서에 작성하여 주십시오.

□ 채점 기준

항목	점수
Policy evaluation 구현	3
Policy Improvement 구현	2
Value Iteration 구현	3
Policy Improvement 구현하여 random policy와 greedy policy와 비교하여 optimal policy 찾기	2

□ 제출기한 및 제출방법

✓ 제출기한

- 2022년 12월 9일 12:59:59 까지 제출

✓ 제출 방법

- 소스코드와 보고서 파일(pdf)을 함께 압축하여 제출

- KLAS -> 과제 제출 -> 압축 파일 제출

✓ 제출 형식

- 파일 이름 : 학번_AI_project2

✓ 보고서 작성 형식

■ 보고서 내용은 한글로 작성

■ 보고서에는 소스코드를 포함하지 않음

■ 채점 기준에 항목은 보고서 내용에 모두 포함되어야 합니다.

■ 아래 각 항목을 모두 포함하여 작성

- Introduction : 프로젝트 내용에 대한 설명

- Algorithm : 프로젝트에서 이용된 Algorithms과 Method의 동작을 설명

- Result : 결과 화면(그래프, 이미지)을 캡처하고 동작을 설명

- Consideration : 고찰 작성