Value iteration and policy iteration

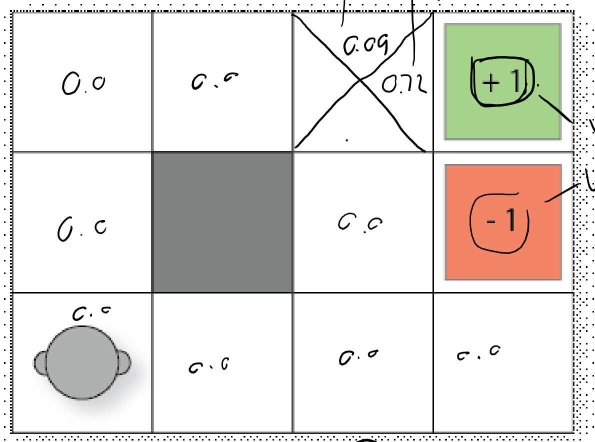
Tuesday, 4 September 2018 9:34 AM

$$V(s)^{\pi} = \max a \in A(s) \sum_{s' \in S} P_a(s'|s)[r(s,a,s') + \gamma V(s')]$$

(moves up) $0 = 0.8(0 + 0.9 \times 0)$
(slips right) $0.09 = 0.1(0 + 0.9 \times 1)$
(slips left) $0 = 0.1(0 + 0.9 \times 0)$
$\Sigma = 0.09$

$0.8(0 + 0.9 \times 1) = \boxed{0.72}$ (moves right)
$+ 0.1(0 + 0.9 \times 0) = 0$ (slips down)
$+ 0.1(0 + 0.9 \times 0) = 0$ (slips up)
$\Sigma = 0.72$

argmax is "right" $\in A$
so $V(s) = 0.72$

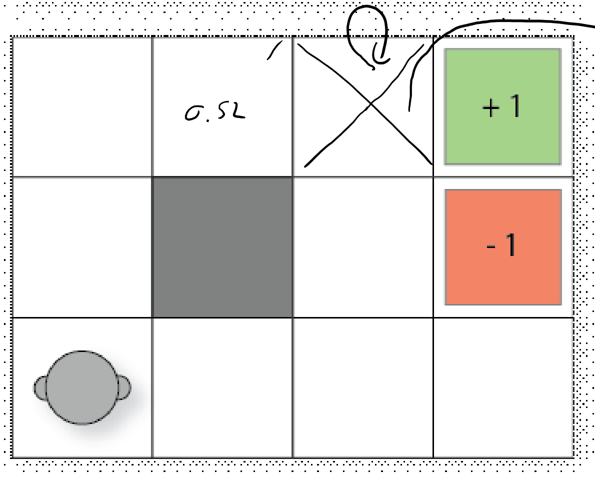


$V(s) = 1$
$V(s) = -1$

The two labelled cells give a reward: 1 and -1 respectively. (Actually, we will assume V(s)=1 or -1)

But! Things can go wrong:

- If the agent tries to move north, 80% of the time, this works as planned (provided the wall is not in the way)
- 10% of the time, trying to move north takes the agent west (provided the wall is not in the way);
- 10% of the time, trying to move north takes the agent east (provided the wall is not in the way)
- If the wall is in the way of the cell that would have been taken, the agent stays put.
- Similar for all other directions



$0.8(0 + 0.9 \times 1)$ 0.72 (moves right)
$+ 0.1(0 + 0.9 \times 0)$ 0 (slips down)
$+ 0.1(0 + 0.9 \times 0.72) = 0.06$ (slips up)
$\Sigma = 0.78$

note: slipping up now receives a reward because it remains in the state

Policy iteration:

1) Start with arbitrary (maybe random) policy pi
2) Calculate V(s) for that policy pi
3) Improve the policy by setting pi(s) := argmax a in A "bellman equation"
4) If pi changes, return to 2, else finish because we have the optimal policy

$0.1(0 + 0.9 \times 1)$