<u>DP – Grid World</u>: Dynamic Programming. *Example obtained from Sutton & Barto, Reinforcement Learning: An Introduction, 2018*

The 5x5 grid in Fig. 1 represents an example of a Markov Decision Process (MDP).

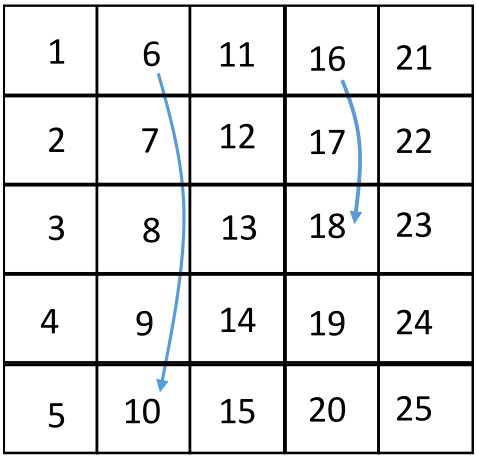

Fig.1

The cells of the grid correspond to the states of the environment, i.e. $\mathcal{S}$ = {1,2,..,25}. At each cell, four actions are possible: north (1), east (2), south (3) and west (4), which deterministically cause the agent to move one cell in the respective direction on the grid, so $\mathcal{A}$ = {1,2,3,4}. Actions that would take the agent off the grid leave its location unchanged, but also result in a reward of -1. Other actions result in a reward of 0, except those that move the agent out of the special states 6 and 16. From state 6, all four actions yield a reward of +10 and take the agent to state 10. From state 16, all actions yield a reward of +5 and take the agent to state 18. See as example the transition graph for states 1, 2 and 6, in Fig. 2, where *r(s,a,s')* is the reward of state *s*, when action *a* is run and the immediate next state is *s*'.

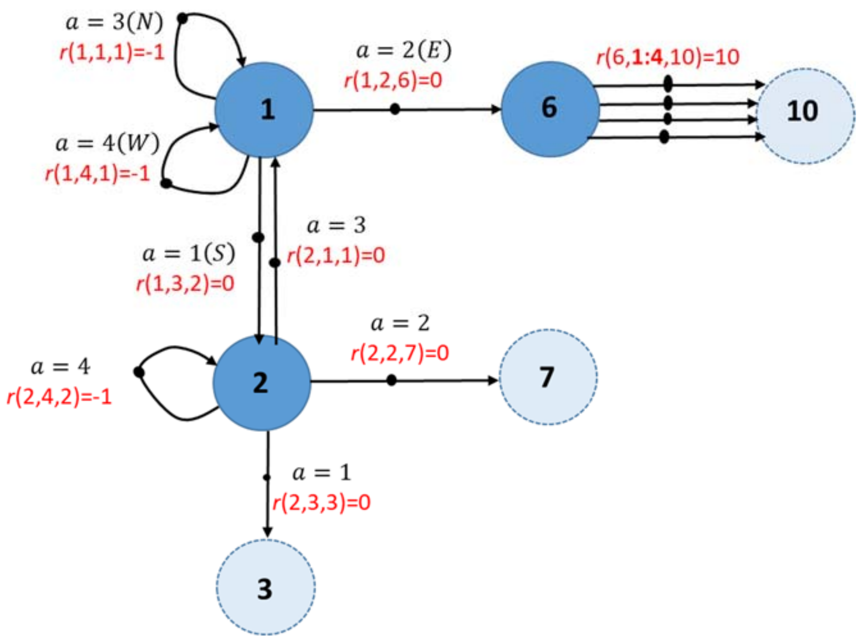

Fig. 2

Use the provided Colab page to solve next questions, and note that in Python we enumerate states from 0 to 24 instead of 1 to 25:

Part I: MARKOV DECISION PROCESS

a) Initiate the variable $p(s'|s,a)$ and the corresponding reward $r(s,a,s')$ by giving values for s,s'=1,..,25 and a=1,..,4.

b) If π is the equiprobable random policy, obtain the reward vector $\mathbf{R}^{\pi}$ and the probability matrix $\mathbf{P}^{\pi}$ shown in the Bellman expectation equation system. Draw in a square image the matrix $\mathbf{P}^{\pi}$ or $p(s'|s)$ taking as axis the states s (vertical axe) and s' (horizontal axe).

c) Solve the Bellman equation with a discount factor γ=0.9 and draw in a square figure (as Fig. 1) the value function of each state s: $v_{\pi}(s)$.

Part II: DYNAMIC PROGRAMING:

d) Program the **Iterative Policy Evaluation** procedure to iteratively compute $v_{\pi}(s)$, for s=1,…,25. Compare the result with the one obtained in question c).

e) Program the **Policy Iteration Improvement procedure** to iteratively compute an optimum deterministic policy $A(s)$, for s=1,…,25, i.e. a policy such as $\pi(a|s)=1$ if $A(s)=a$ and $\pi(a'|s)=0$ for a'<>a. Draw in square figures (as Fig. 1) the value function of each state s: $v_A(s)$ and the optimum policy $A(s)$. You can make use of the iterative policy_evaluation function programmed in d). Add comments on the number of iterations, the θ value used. Compare $v_A(s)$ with $v_{\pi}(s)$ obtained in question c).

f) Program the **Value Iteration Improvement procedure** to iteratively compute an optimum deterministic policy $A(s)$, for s=1,…,25, i.e. a policy such as $\pi(a|s)=1$ if $A(s)=a$ and $\pi(a'|s)=0$ for a'<>a. Draw in square figures (as Fig. 1) the value function of each state s: $v_A(s)$ and the optimum policy $A(s)$. Add comments on the number of iterations, the θ value used. Compare $v_A(s)$ with $v_{\pi}(s)$ obtained in questions c) and e).