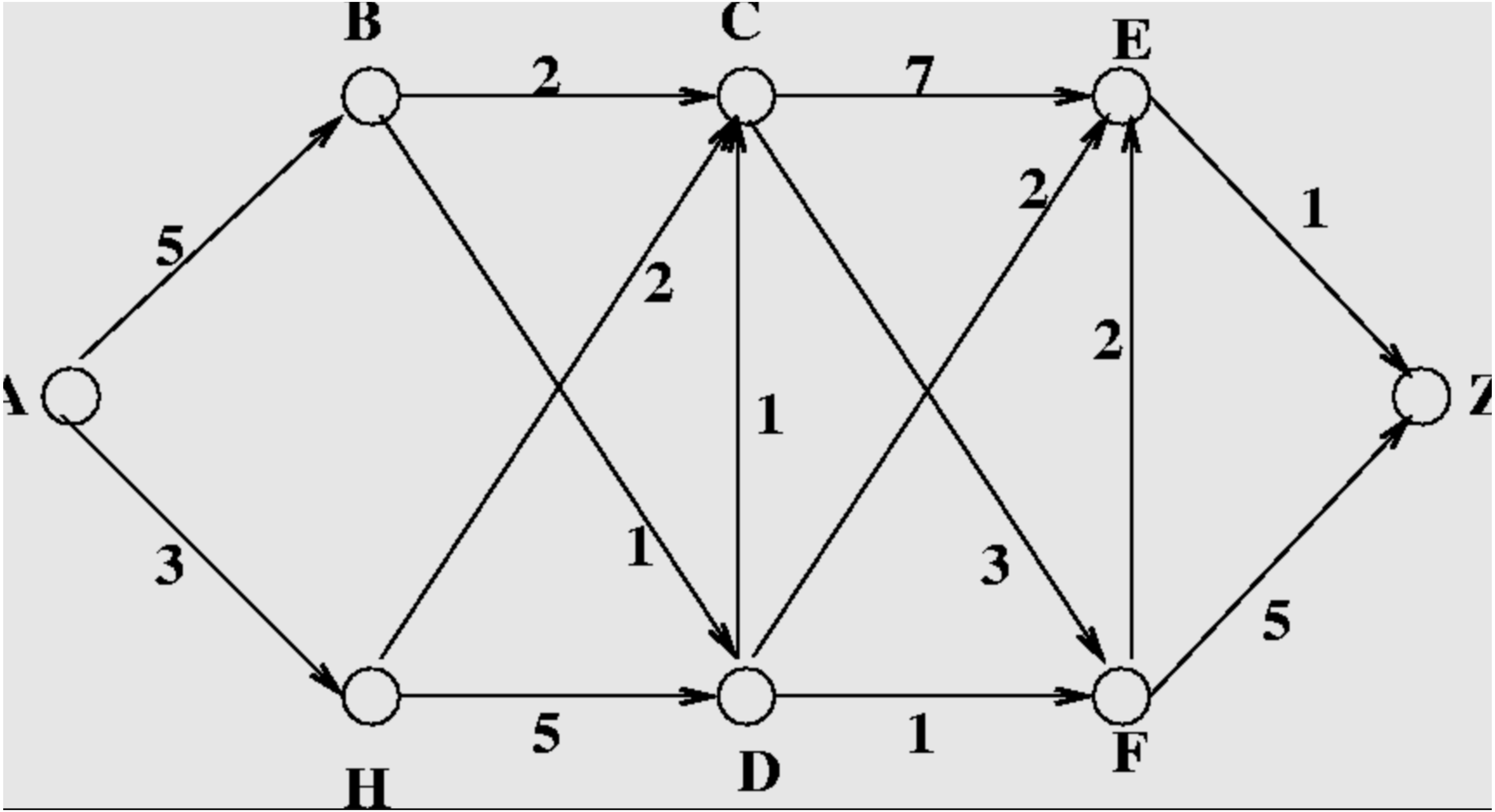

Find the shortest path from A to Z using dynamic programming. Show your work.

Easy to see that 6 is an upper bound on the length of the path, so we can solve a dynamic programming problem with K=6:

A	B	C	D	E	F	H	Z	
Infinity	Infinity	Infinity	Infinity	1	5	Infinity	0	Time step 5
Infinity	Infinity	8	3	1	3	Infinity	0	Time step 4
Infinity	4	6	3	1	3	8	0	Time step 3
9	4	6	3	1	3	8	0	Time step 2
9	4	6	3	1	3	8	0	Time step 1

Keeping track of which action gives the maximum reward, we get that the shortest path is ABDEZ.

Consider an MDP with two states, A and B. In A, there are two actions you can take. Action 1 keeps you in state A, with a reward of one. Action 2 moves you to B, with a reward of zero. In state B, there is only one action to take, which keeps you in B with a reward of 2.

(i) What is the smallest value of the discount factor for which, starting in A, moving to B is optimal?

Staying in state A brings a reward of $1 + \gamma + \gamma^2 + \cdots = 1/(1-\gamma)$

Moving to B brings a reward of $0 + 2\gamma + 2\gamma^2 + \cdots = 2\gamma\dfrac{1}{1-\gamma}$

So we need $2\gamma \geq 1$ or $\gamma \geq 1/2$

Consider an MDP with two states, A and B. In A, there are two actions you can take. Action 1 keeps you in state A, with a reward of one. Action 2 moves you to B, with a reward of zero. In state B, there is only one action to take, which keeps you in B with a reward of 2.

(ii) Consider the policy which takes a random action. Perform the first two iterations of policy evaluation starting from [16,16] with a discount factor of 1/2. Do this by hand (i.e., do not write code).

$$V_1 = \begin{pmatrix} (1/2)\cdot 1 + (1/2)\cdot 0 + \frac{1}{2}\frac{1}{2}16 + \frac{1}{2}\frac{1}{2}16 \\ 2 + 1\cdot\frac{1}{2}16 \end{pmatrix} = \begin{pmatrix} 8.5 \\ 10 \end{pmatrix}$$

$$V_2 = \begin{pmatrix} (1/2)\cdot 1 + 1/2\cdot 0 + \frac{1}{2}\frac{1}{2}8.5 + \frac{1}{2}\frac{1}{2}10 \\ 2 + 1\cdot\frac{1}{2}10 \end{pmatrix} = \begin{pmatrix} 5.125 \\ 7 \end{pmatrix}$$

B, with a reward of zero. In state B, there is only one action to take, which keeps you in B with a reward of 2.

(iii) Perform (again by hand) the first two iterations of value iteration for this problem, also starting from [16,16] with discount of 1/2.

$$V_1 = \begin{pmatrix} \max(1+\gamma 16, 0+\gamma 16) \\ 2+\gamma 16 \end{pmatrix} = \begin{pmatrix} 9 \\ 10 \end{pmatrix}$$

$$V_2 = \begin{pmatrix} \max(1+\gamma 9, 0+\gamma 10) \\ 2+\gamma 10 \end{pmatrix} = \begin{pmatrix} 5.5 \\ 7 \end{pmatrix}$$

- Suppose you add a constant to all costs in an MDP. Is it always true that the set of optimal policies remains the same?
  Give an answer to this question in the cases when:
  (i) the MDP is continuing.
  (ii) the MDP is terminal.
  You can assume that, in both cases, the discount factor is strictly less than one.
- **Solution:** In case (i), the set of optimal policies remains the same. Indeed, if c is added to every reward, then every policy incurs an extra cost of $c + \gamma c + \gamma^2 c + \cdots = \dfrac{c}{1-\gamma}$
- In case (ii), the set of optimal policies can change. Consider, for example, an MDP with a single terminal state with a -1 reward on all transitions. The best policy will reach the terminal state the soonest.

  But if we add two to every reward, so the the reward on each transition is $+1$, then the optimal policy is the one which reaches the terminal state the slowest.

- Suppose someone gives you a randomized policy π in a dynamic programming problem and claims it is optimal. How would you construct a deterministic policy which is optimal? Justify your answer.
- Suppose the dynamic programming policy takes K steps. Let us introduce the notation $\pi_k(s)$ to be the distribution of actions given by the randomized optimal policy π at step k at state s.

  At step $K-1$, for each state s, the randomized optimal policy $\pi_{K-1}(s)$ can only give nonzero probabilities to actions maximizing $r(s,a) + \gamma \sum_{s'} P(s \mid s,a) J_K(s')$ (by the same argument in the lecture notes).
- So as a first step to obtain a deterministic optimal policy, we simply pick any action a such that $\pi_{K-1}(a \mid s) > 0$ and define a new policy which always takes this action a in state s at time $K-1$.

- Now consider step $K-2$. By the same argument as in the lecture notes on dynamic programming, for each state s, the randomized optimal policy $\pi_{K-2}(s)$ can only give nonzero probabilities to actions maxizing $r(s,a)+\gamma\sum_{s'}P(s'\,|\,s,a)J_{K-1}(s')$
- So to obtain a deterministic optimal policy, we simply pick any action a such that $\pi_{K-2}(a\,|\,s)>0$ and define a new policy which always takes this action a in state s at time K_2.
- We can recurse this way to get to any time step t. The answer is, at any time step, to pick one action a to which the randomized optimal policy assigns a positive probability, and define a deterministic policy which always chooses that action.