Homework 2 - Probabilistic aspects of computer science

1 The maximal expected reward

Let X_i denote the random state at time i and Y_i denote the random action at time i of an MDP. Given a policy $\boldsymbol{\pi}$, the *maximal expected reward* at time horizon t of $\boldsymbol{\pi}$ is defined by:

$$M_t^{\boldsymbol{\pi}} \stackrel{\text{def}}{=} \mathbf{E}^{\boldsymbol{\pi}} \left(\max(r(X_i, Y_i) \mid 0 \leq i < t)\right)$$

The corresponding vectorial reward (which depends on the initial state) is denoted $\mathbf{M}_t^{\boldsymbol{\pi}}$. As usual, the optimal vectorial reward $\mathbf{M}_t^*$ is defined by: for all $s \in S$, $\mathbf{M}_t^*[s] \stackrel{\text{def}}{=} \sup_{\boldsymbol{\pi}}(\mathbf{M}_t^{\boldsymbol{\pi}}[s])$.

Question 1. Show an example of MDP such that no Markovian policy is optimal for the (vectorial) maximal expected reward at time horizon 3.

Question 2. Let $\mathcal{M}$ be an MDP and t be an horizon. Propose an algorithm that finds the optimal reward and an optimal policy for the maximal expected reward problem in polynomial time w.r.t. the size of $\mathcal{M}$ and in pseudo-polynomial time w.r.t. t.
Hint: The algorithm builds an MDP $\mathcal{M}'$ such that from the optimal reward and an optimal policy for the pure total expected reward in $\mathcal{M}'$, one can recover the optimal reward and an optimal policy for the maximal expected reward in $\mathcal{M}$.

2 Terminal components of a MDP

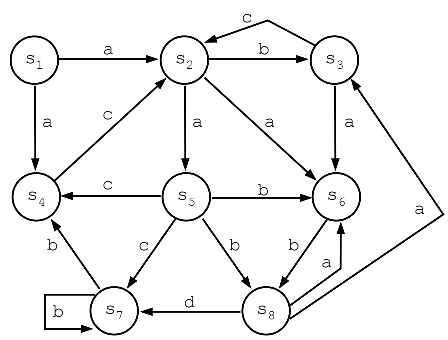
Let $\mathcal{M}$ be an MDP, we introduce the notion of a subMDP. A subMDP $\mathcal{M}'$ of $\mathcal{M}$ is a *non empty* set of pairs state-action such that $(s, a) \in \mathcal{M}'$ implies that $s \in S$ and $a \in A_s$. The underlying graph of $\mathcal{M}'$, $G_{\mathcal{M}'} = (S', E')$ is defined by:

1. $S' \stackrel{\text{def}}{=} \{s \in S \mid \exists (s, a) \in \mathcal{M}'\}$;
2. $E' \stackrel{\text{def}}{=} \{(s, s') \in (S')^2 \mid \exists (s, a) \in \mathcal{M}' \text{ with } p(s'|s, a) > 0\}$.

A subMDP$\mathcal{M}'$ is a *terminal component* of $\mathcal{M}$ if:

1. For all $s, s' \in S$, $a \in A_s$, $(s, a) \in \mathcal{M}'$ and $p(s'|s, a) > 0$ implies $s' \in S'$;
2. $G_{\mathcal{M}'}$ is strongly connected.

$\mathcal{M}'$, a terminal component of $\mathcal{M}$, is *maximal* if there is no terminal component $\mathcal{M}''$ with $S' \subseteq S''$, $E' \subseteq E''$ and $S' \cup E' \subsetneq S'' \cup E''$.



We have drawn above $G_{\mathcal{M}}$ the underlying graph of a MDP $\mathcal{M}$ where an action a labels an edge (s, s') if $p(s'|s, a) > 0$.

Question 3. Let $\mathcal{M}$ be the MDP whose graph is drawn above. Find a maximal terminal component of $\mathcal{M}$ and a non maximal terminal component of $\mathcal{M}$.

Let $\rho = s_0, a_0, s_1, a_1, \ldots$ be an infinite path. Define $\omega(\rho) \stackrel{\text{def}}{=} \{(s, a) \mid \forall i \in \mathbb{N}\ \exists j \geq i\ (s_j, a_j) = (s, a)\}$, the set of pairs state-action infinitely occurring in ρ.

Question 4. Let $\boldsymbol{\pi}$ be a policy and $\rho = X_0, Y_0, X_1, Y_1, \ldots$ the random path of an MDP. Prove that:

$$\mathbf{Pr}^{\boldsymbol{\pi}}(\omega(\rho) \text{ is a terminal component}) = 1$$

Algorithm 1: Computing the maximal terminal components

```
MaxTerminalComponents(𝓜)
Input: 𝓜, an MDP
Output: 𝓢𝓜, the set of maximal terminal components of 𝓜
Data: i integer, s, s' states, a action, sub, sub' subMDP, stack, a stack of subMDP
sub ← {(s,a) | s ∈ S, a ∈ A_s}; Push(stack, sub); 𝓢𝓜 ← ∅
while not Empty(stack) do
    sub ← Pop(stack); S' ← {s | ∃(s,a) ∈ sub}
    for (s,a) ∈ sub do
        for s' ∈ S do
            if p(s'|s,a) > 0 and s' ∉ S' then sub ← sub \ {(s,a)}
        end
    end
    if sub ≠ ∅ then
        Compute the strongly connected components of G_sub, S_1, ..., S_K
        if K > 1 then
            for i from 1 to K do sub' ← {(s,a) ∈ sub | s ∈ S_i}; Push(stack, sub')
        else 𝓢𝓜 ← 𝓢𝓜 ∪ sub
    end
end
return 𝓢𝓜
```

Question 5. Prove that algorithm 1 returns the set of maximal terminal components.

Question 6. Analyse the (worst-case) complexity of algorithm 1 w.r.t. $|S|$ and $|A|$.

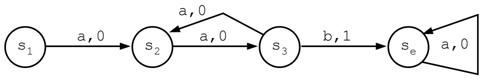
3 Minimising the reachability cost

Let $\mathcal{M}$ be an MDP with *non negative rewards* and an *absorbing state* s_e: A_{s_e} is a singleton whose Dirac distribution leads to s_e and whose reward is null. We assume that there exist policies that ensure to reach s_e with probability 1 and such policies are called *winning policies*. In this case, there exists a stationary deterministic winning policy.
The reachability cost of a policy $\boldsymbol{\pi}$ (which may be infinite) is defined by:

$$R^{\boldsymbol{\pi}} \stackrel{\text{def}}{=} \sum_{i \in \mathbb{N}} \mathbf{E}^{\boldsymbol{\pi}}(r(X_i, Y_i))$$

The corresponding vectorial cost (which depends on the initial state) is denoted $\mathbf{R}^{\boldsymbol{\pi}}$. The optimal vectorial cost $\mathbf{R}^*$ is defined by: for all $s \in S$, $\mathbf{R}^*[s] \stackrel{\text{def}}{=} \inf_{\boldsymbol{\pi}}(\mathbf{R}^{\boldsymbol{\pi}}[s] \mid \boldsymbol{\pi} \text{ is winning})$. The reachability cost problem consists to find the minimal reachability cost $\mathbf{R}^*$ and an optimal winning policy.

Question 7. Using the MDP figured below (with only Dirac distributions) show that a non winning strategy can have a smaller reachability cost than any winning strategy.



In the sequel, we assume that for all non winning policy $\boldsymbol{\pi}$ there exists $s \in S$ such that: $\mathbf{R}^{\boldsymbol{\pi}}[s] = \infty$.
Let the operator L on $Rew \stackrel{\text{def}}{=} \{\mathbf{v} \in \mathbb{R}^S \mid \mathbf{v}[s_e] = 0 \wedge \forall s \in S\ \mathbf{v}[s] \geq 0\}$ be defined by:

$$\forall s \in S\ L(\mathbf{v})[s] = \min_{a \in A_s} \left(r(s,a) + \sum_{s' \in S} p(s'|s,a)\mathbf{v}[s'] \right)$$

Question 8. Let $\mathbf{v} \in Rew$ be a fixpoint of L. Prove that $\mathbf{v} \leq \mathbf{R}^*$.

Question 9. Let d^∞ be a stationary policy. Show that $\mathbf{R}^{d^\infty} = \sum_{i \in \mathbb{N}} (\mathbf{P}_d)^i \mathbf{r}_d$ (using the notations of the lecture notes).

Let d^∞ be a winning policy. Since d^∞ is stationary, one can build an ordering of $S = \{s_1, \ldots, s_n\}$ such that $s_1 = s_e$ and for all s_i with $i > 1$ there exists $\alpha_i < i$ such that $\mathbf{P}_d[i, \alpha_i] > 0$. Let $p = \min(\min(\mathbf{P}_d[i, \alpha_i] \mid i > 1), \frac{1}{2})$. Define $\mathbf{v} \in Rew$ by $\mathbf{v}[s_i] = 1 - p^{2i}$ for $i > 1$.

Question 10. Show that $\mathbf{P}^{d^\infty}\mathbf{v} \leq \gamma \mathbf{v}$ with $\gamma \stackrel{\text{def}}{=} \frac{1-p^{2n-1}}{1-p^{2n}}$. Deduce that $\mathbf{R}^{d^\infty}$ is finite.

Given d a decision rule, the operator L_d on Rew is defined by:

$$L_d(\mathbf{v}) = \mathbf{r}_d + \mathbf{P}_d \mathbf{v}$$

Question 11. Let d^∞ be a winning policy. Show that $\mathbf{R}^{d^\infty}$ is a fixpoint of L_d.

Question 12. Let d be a decision rule such that there exists $\mathbf{v} \in Rew$ with $L_d(\mathbf{v}) \leq \mathbf{v}$.
Show that d^∞ is a winning policy. *Hint: use the assumption about non winning policies.*

Question 13. Let d^∞ be a deterministic stationary winning policy such that $L(\mathbf{R}^{d^\infty}) \lneq \mathbf{R}^{d^\infty}$.
Let d' be a deterministic decision rule such that $L(\mathbf{R}^{d^\infty}) = L_{d'}(\mathbf{R}^{d^\infty})$.
Show that $\mathbf{R}^{d'^\infty} \lneq \mathbf{R}^{d^\infty}$.

Question 14. Deduce from the previous questions that there exists a winning deterministic stationary policy d^∞ such that $L(\mathbf{R}^{d^\infty}) = \mathbf{R}^{d^\infty}$ and that d^∞ is an optimal policy for the reachability cost problem.

Question 15. Design a linear programming problem such that its solution is $\mathbf{R}^*$.