Question 1. (5 points)

Agent receives rewards as follows:

$$r_k = \begin{cases} 0; \text{ if } y_k = 0 \\ 1; \text{ if } y_k = 1 \text{ and } k = 1 \\ 2r_{k-1}\text{with probability}\frac{1}{2}; \text{ if } y_k = 1 \text{ and } k > 1 \\ \frac{1}{2}r_{k-1}\text{with probability}\frac{1}{2}; \text{ if } y_k = 1 \text{ and } k > 1 \end{cases} \tag{1}$$

Determine $U^{y_{\leq 3}}$ of $y_1 = y_2 = y_3 = 1$.

Answer:

There are four possible reward sequences $r_{\leq 3}$ each with probability $\frac{1}{4}$:

$$\begin{gathered} 1, \tfrac{1}{2}, \tfrac{1}{4} \\ 1, \tfrac{1}{2}, 1 \\ 1, 2, 1 \\ 1, 2, 4 \end{gathered}$$

Their respective sums are $\frac{7}{4}, \frac{10}{4}, \frac{16}{4}, \frac{28}{4}$. The mean value is thus $\frac{1}{4}(\frac{7}{4} + \frac{10}{4} + \frac{16}{4} + \frac{28}{4}) = \frac{61}{16}$

Question 2. (10 points)

Agent receives rewards as follows:

$$r_k = \begin{cases} 0; \text{ if } y_k = 0 \\ 1; \text{ if } y_k = 1 \text{ and } k = 1 \\ r_{k-1} + 1 \text{ with probability}\frac{1}{2}; \text{ if } y_k = 1 \text{ and } k > 1 \\ r_{k-1} - 1 \text{ with probability}\frac{1}{2}; \text{ if } y_k = 1 \text{ and } k > 1 \end{cases} \tag{2}$$

Determine $U^{y_{\leq\infty}}$ of $y_{\leq\infty} = 1, 1, \ldots$ for $\gamma = \frac{1}{2}$.

Answer:

Consider the conditional expected value of reward r_k in a reward sequence $r_{\leq m}$ of length $m \geq k$

$$\mathbb{E}\left(r_k \mid y_{\leq\infty} = 1, 1, \ldots\right) = \sum_{r_{\leq m}} P(r_{\leq m} | y_{\leq\infty} = 1, 1, \ldots) r_k$$

Since the condition $y_{\leq\infty} = 1, 1, \ldots$ is fixed by assumption, we drop it from the probability and expectation expressions. Due to the second case in (2), we have $\mathbb{E}\left(r_1\right) = 1$, and from the last two cases we have $\mathbb{E}\left(r_k\right) = \mathbb{E}\left(r_{k-1}\right)$ for $k \geq 2$. So by induction

$$\mathbb{E}\left(r_k\right) = \sum_{r_{\leq m}} P(r_{\leq m}) r_k = 1 \tag{3}$$

for all $k \in \mathbb{N}$. The utility can now be computed as follows:

$$U^{y_{\leq\infty}} = \lim_{m\to\infty} \sum_{r_{\leq m}} \left(P(r_{\leq m}) \sum_{k=1}^{m} r_k \gamma^{k-1} \right) = \lim_{m\to\infty} \sum_{k=1}^{m} \gamma^{k-1} \underbrace{\sum_{r_{\leq m}} P(r_{\leq m}) r_k}_{=1 \text{ from } (3)} = \lim_{m\to\infty} \sum_{k=1}^{m} \left(\frac{1}{2}\right)^{k-1} = 2$$

Question 3. (5 points)

Consider classification with $Y = \{ 0, 1 \}$, where $P_c(y^*|x)$ is the probability that y^* is the true class of x, and rewards are given as

$$r_k = \begin{cases} 0 \text{ if } y = y^* \\ -1 \text{ if } y = 0 \text{ and } y^* = 1 \\ -3 \text{ if } y = 1 \text{ and } y^* = 0 \end{cases}$$

Consider the policy

$$y(x) = \arg\max_y P_c(y|x)$$

is this policy necessarily optimal, i.e. does it always coincide with the policy

$$\overline{\pi}(x) = \arg\max_y \mathbb{E}\left(r \mid x, y\right)$$

? Justify your answer mathematically.

Answer:

No. Consider an observation x such that $P_c(0|x) = 1/3$ and $P_c(1|x) = 2/3$

$$\begin{aligned}\mathbb{E}\left(r \mid x, 0\right) &= -1 \cdot P(-1|x,0) - 3 \cdot P(-3|x,0) = -1 \cdot P_c(1|x) - 3 \cdot 0 = -2/3\\ \mathbb{E}\left(r \mid x, 1\right) &= -1 \cdot P(-1|x,1) - 3 \cdot P(-3|x,1) = -1 \cdot 0 - 3 \cdot P_c(0|x)) = -1\end{aligned}$$

So

$$0 = \arg\max_y \mathbb{E}\left(r \mid x, y\right) \neq \arg\max_y P_c(y|x) = 1$$

Question 4. (2 points)

Discuss how the *exploration-exploitation* dilemma manifests itself in the *concept learning* scenario. Specify the conditions on which the execution of random actions would (would not) be useful for a concept-learning agent.

Answer:

In concept learning (as defined in the lectures), the reward r for x, y is known with certainty on the first execution of y one step after receiving x so getting more reward samples for that pair is useless for estimating $P(r|x,y)$. However, without further assumptions, future observation may depend on the current reward so random (non-optimal) actions could be used to explore such a dependence.

Consider e.g. a "husband agent" told by "wife environment": $x =$"wash the dishes". The agent does not like to receive such x because the rewards for acting on such x (which include time spent, missing other joys, ..) are usually small. A possible strategy is to break all the dishes during wash-up, receiving a very low r this time but with the chance of never receiving x again.

On the other hand, when x are sampled i.i.d., so they do not depend on the history, exploration through randomized actions is pointless.

Question 5. (2 points)

Consider an algorithm that learns *monotone* disjunctions (or monotone conjunctions) from n-tuples of Boolean attribute values corresponding to n propositional variables. How can you use that algorithm to learn *general* disjunctions (or conjunctions) without changing it? You may change the number of inputs. How will your solution change the mistake bound in the case of the Winnow algorithm? Consider the number s of literals in the target disjunction constant.

Answer:

By basis expansion: introduce additional n attributes holding the inverted values of the original n attributes, thus converting the task to learning a monotone disjunction on $2n$ variables from $2n$-tuples.

For Winnow, the bound

$$2 + 2s \log n$$

changes to

$$2 + 2s \log 2n = 2 + 2s \log 2n = 2 + 2s(1 + \log n) = 2 + 2s \log n + 2s$$

i.e., only by the additive constant $2s$.

Question 6. (1 points)

Let h, h' be propositional conjunctions. Is $h' \models h$ equivalent to $h \subseteq h'$? Justify your answer.

Answer:

The two relations are not equivalent. For example $\mathrm{p} \wedge \neg\mathrm{p} \models \mathrm{q}$ but $\mathrm{q} \not\subseteq \mathrm{p} \wedge \neg\mathrm{p}$. The equivalence would hold if h' was not tautologically false.

Question 7. (4 points)

Let h, h' be contingent propositional conjunctions that prescribe policies by

$$y = h(x) = 1 \text{ iff } x \models h$$

We say that h at least as general as h' if $h(x) = 1$ for any $x \in X$ such that $h'(x) = 1$. Is it true that $h' \models h$ if and only if h is at least as general as h'? Justify your answer.

Answer:

No. Consider

$$\begin{aligned} h &= \mathrm{p} \\ h' &= \mathrm{q} \\ X &= \{\, \mathrm{p} \wedge \mathrm{q} \,\} \\ h(\mathrm{p} \wedge \mathrm{q}) &= h'(\mathrm{p} \wedge \mathrm{q}) = 1 \end{aligned}$$

h is as general as h' but $h' \not\models h$.

Tautological consequence does not depend on an interpretation domain, while the generality relation depends on the observation set X.

Question 8. (15 points)

Consider an algorithm learning in the mistake bound model. Prove that if the condition $\sum_{k=1}^{\infty} |r_k| \leq \mathit{poly}(n_X)$ $(n_X \in \mathbb{N})$ is satisfied, then from some time $K \in \mathbb{N}$ the agent will not make any mistakes, i.e.,

$$\exists K \in \mathbb{N}, \forall k \in \mathbb{N} : k > K \rightarrow r_k = 0$$

Answer:

The assumption $\sum_{k=1}^{\infty} |r_k| \leq \mathit{poly}(n_X)$ $(n_X \in \mathbb{N})$ implies that $\sum_{k=1}^{\infty} |r_k|$ is convergent, i.e., (4) below. An elementary calculus lemma says that a series with non-negative terms is convergent iff it is upper-bounded. We only need to prove the 'only if' direction of it, i.e.,

$$\lim_{m\to\infty} \sum_{k=1}^{m} |r_k| < \infty \tag{4}$$

$$\Rightarrow$$

$$\exists b \in \mathbb{R}, \forall m \in \mathbb{N} : \sum_{k=1}^{m} |r_k| \leq b \tag{5}$$

Let $s_m = \sum_{k=1}^{m} |r_k|$ and $l = \lim_{m\to\infty} s_m$. From the definition of limit, $\forall \epsilon > 0, \exists N_\epsilon \in \mathbb{N}$ such that $\forall m > N_\epsilon : |s_m - l| < \epsilon$. Let $N = N_\epsilon$ for $\epsilon = 1$. Then

$$\forall m > N_\epsilon : |s_m - l| < 1$$

i.e.,

$$\forall m > N_\epsilon : l - 1 < s_m < l + 1$$

Since $l < \infty$ due to (4), the infinite set $\{ s_{N+1}, s_{N+2}, \dots \}$ is (lower and upper) bounded. Also the set $\{ s_1, s_2, \dots s_N \}$ is bounded because it is finite. So their union, i.e. $\{ s_1, s_2, \dots \}$ is bounded, thus $\forall m \in \mathbb{N} : s_m \leq b$ where $b \in \mathbb{R}$ is an upper bound. Therefore (5) holds. From (5) it follows that there is only a finite set of indexes k such that $|r_k| > 0$. Let K be the maximum of this finite set and we finally have $\forall k \in \mathbb{N} : k > K \rightarrow r_k = 0$.

Intuitively, one could admit that (4) is true even with an infinite number of mistakes $|r_k| \neq 0$ as long as the 'spaces' between the non-zero rewards in the k-indexed series are progressively larger, and the size of these spaces increases sufficiently quickly to guarantee convergence. The proof above shows that under the standard axiomatization of calculus, this is not the case and there indeed is a finite time K after which no mistakes are made.

Question 9. (3 points)

Give two examples of a non-contingent conjunction, one tautologically true and one tautologically false. Do the same for disjunctions. Explain how an incomplete truth assignment to n propositional variables is represented by a conjunction and decide whether such a conjunction may be tautologically false. Explain why Winnow does not learn from incomplete truth assignments.

Answer:

	true	false
conjunction	$\emptyset$	$\mathrm{p} \wedge \neg \mathrm{p}$
disjunction	$\mathrm{p} \vee \neg \mathrm{p}$	$\emptyset$

Let $\mathrm{p}_1, \mathrm{p}_2, \dots, \mathrm{p}_n$ be propositional variables assigned the respective truth-values by observations $x = x^1, x^2, \dots, x^n$. A conjunction representing x includes literal p_i ($\neg \mathrm{p}_i$, respectively) iff $x^i = 1$ ($x^i = 0$). The two cases are mutually exclusive so a variable cannot be included both with and without negation, thus the conjunction cannot be tautologically false.

Winnow decision policy computes the dot product $h \cdot x$ for which the entire tuple x must be provided.

Question 10. (5 points)

Show where the assumption that the target hypothesis is a *monotone disjunction* is needed in the proof of Winnow mistake bound and explain how the proof would fail if assuming a general target concept on $X = \{ 0, 1 \}^n$.

Answer:

The proof says:

> *Each promotion doubles h^i where i is one of the s indexes corresponding to the s atoms in the target disjunction.*

This would not be necessarily true if the target hypothesis $\overline{h}$ was not a monotone disjunction. Say $n = 2$, $x = (0, 1)$ and the target hypothesis $\overline{h} = \neg \mathrm{p}_1$ is a *non-monotone* disjunction. Assume the agent's hypothesis is $h = (1, 1)$, so x is a false negative, thus h gets promoted. By the Winnow learning rule, only the component h^2 is doubled towards $(1, 2)$ but p_2 is not in the target disjunction.

The proof also deals with the marginal case

> *If on the other hand $s = 0$ then the target disjunction is tautologically false.*

This would also not be correct if the target hypothesis was e.g. a conjunction (the empty conjunction is tautologically true.)

Question 11. (2 points)

Give the lgg for all pairs from

$$\mathcal{H} = \{\, \mathrm{p} \wedge \mathrm{q}, \mathrm{p} \wedge \neg\mathrm{q}, \mathrm{p} \vee \mathrm{q}, \mathrm{p} \vee \neg\mathrm{q} \,\}$$

whenever the lgg is defined for the pair.

Answer:

We defined lgg for a pair of conjunctions or a pair of disjunctions. For all $h \in \mathcal{H}$, the $\mathsf{lgg}(h, h) = h$. Then,

$$\mathsf{lgg}(\mathrm{p} \wedge \mathrm{q}, \mathrm{p} \wedge \neg\mathrm{q}) = \mathsf{lgg}(\mathrm{p} \vee \mathrm{q}, \mathrm{p} \vee \neg\mathrm{q}) = \mathrm{p}$$

and since lgg is commutative, the remaining two defined pairs (flipping the arguments above) are also p.

Question 12. (3 points)

In concept classification, the generalization algorithm receives the sequence

$$x_1, x_2, \ldots, x_{10}$$

of observations (contingent conjunctions) where all x_k with odd indexes $(x_1, x_3, \ldots)$ are positive examples, and all the others are negative. The agent's sequence of hypotheses is

$$h_1, h_2, \ldots, h_{10}$$

so if some hypothesis is unchanged for m time steps, then there is a subsequence of m identical hypotheses above.

Determine

1. whether $h_2 = x_1$ or $h_2 = x_2$ or none of these options;
2. the sequence of hypotheses for the same agent that receives observations

$$x_1, x_1, x_3, x_3, \ldots, x_9, x_9$$

Answer:

1. $h_2 = x_1$. By the description given in the lectures, the first mistake is made on the first positive example x_1, so $r_2 = -1$, at which time the hypothesis is updated so $h_2 = x_1$.
2. The sequence of hypotheses is the same as for the former agent because $h_{k+1} = h_k$ both if x_k is a negative example (as in every second observation in the former case) or if $h_k \subseteq x_k$ (as in every second observation in the latter case).

Question 13. (10 points)

Let h, h' be two propositional clauses or conjunctions. Show that $\mathsf{lgg}(h, h') = \mathsf{Lits}(h) \cap \mathsf{Lits}(h')$ is a least general generalization of h, h'.

Answer:

For shorter notation, we apply set operations on clauses or conjunctions as if they were sets of literals, so for example $\mathcal{L} \in h$ means $\mathcal{L} \in \mathsf{Lits}(h)$. The question assumes that $g = \mathsf{lgg}(h, h') = h \cap h'$, i.e.,

$$\forall \mathcal{L} : \mathcal{L} \in g \leftrightarrow \mathcal{L} \in h \wedge \mathcal{L} \in h' \tag{6}$$

from which we deduce

$$\vdash \forall \mathcal{L} : \mathcal{L} \in g \to \mathcal{L} \in h \wedge \mathcal{L} \in h'$$
$$\vdash \forall \mathcal{L} : \mathcal{L} \in g \to \mathcal{L} \in h \text{ and } \vdash \forall \mathcal{L} : \mathcal{L} \in g \to \mathcal{L} \in h' \tag{7}$$

The two formulas we deduced in (7) define the respective subset relations $g \subseteq h$ and $g \subseteq h'$ which we were to prove according to the definition of least general generalization. In addition, we are supposed to prove that there is no g' such that $g \subset g'$, $g' \subseteq h$, $g' \subseteq h'$. Assume for contradiction, that such a g' exists, i.e.,

$$\forall \mathcal{L} : \mathcal{L} \in g \rightarrow \mathcal{L} \in g' \tag{8}$$
$$\exists \mathcal{L} : \mathcal{L} \in g' \wedge \neg(\mathcal{L} \in g) \tag{9}$$
$$\forall \mathcal{L} : \mathcal{L} \in g' \rightarrow \mathcal{L} \in h \tag{10}$$
$$\forall \mathcal{L} : \mathcal{L} \in g' \rightarrow \mathcal{L} \in h' \tag{11}$$

We rewrite formulas 8 - 11 as clauses, *Skolemizing* formula 9.

$$\mathcal{L} \in g \rightarrow \mathcal{L} \in g' \tag{12}$$
$$\mathsf{a} \in g' \tag{13}$$
$$\neg(\mathsf{a} \in g) \tag{14}$$
$$\mathcal{L} \in g' \rightarrow \mathcal{L} \in h \tag{15}$$
$$\mathcal{L} \in g' \rightarrow \mathcal{L} \in h' \tag{16}$$

We also rewrite assumption (6) as a clause set

$$\mathcal{L} \in g \rightarrow \mathcal{L} \in h \tag{17}$$
$$\mathcal{L} \in g \rightarrow \mathcal{L} \in h' \tag{18}$$
$$\mathcal{L} \in h \wedge \mathcal{L} \in h' \rightarrow \mathcal{L} \in g \tag{19}$$

We are supposed to find a contradiction in the clause set 12 - 19. We proceed by resolution:

- Resolve (15) ($\mathcal{L} \mapsto \mathsf{a}$) with (13) obtaining
$$\mathsf{a} \in h \tag{20}$$
- Resolve (16) ($\mathcal{L} \mapsto \mathsf{a}$) with (13) obtaining
$$\mathsf{a} \in h' \tag{21}$$
- Resolve (19) ($\mathcal{L} \mapsto \mathsf{a}$) with (20) obtaining
$$\mathsf{a} \in h' \rightarrow \mathsf{a} \in g \tag{22}$$
- Resolve (22) with (21) obtaining
$$\mathsf{a} \in g \tag{23}$$
- Resolve (23) with (14) obtaining the contradiction $\square$.

Question 14. (15 points)

Determine if

1. $h \subseteq_\theta h'$
2. $h' \models h$

for

$$h = \mathrm{p}(x, y) \wedge \mathrm{p}(y, z) \wedge \neg\mathrm{p}(x, z)$$
$$h' = \mathrm{p}(\mathsf{a}, \mathsf{b}) \wedge \mathrm{p}(\mathsf{b}, \mathsf{c}) \wedge \mathrm{p}(\mathsf{c}, \mathsf{d}) \wedge \neg\mathrm{p}(\mathsf{a}, \mathsf{d})$$

Answer:

1. $h \subseteq_\theta h'$ does not hold because for no substitution θ, $\mathsf{Lits}(h\theta) \subseteq \mathsf{Lits}(h')$. Indeed, for $\neg \mathrm{p}(x,z)\theta$ to be in $\mathsf{Lits}(h')$, necessarily $\{\, x \mapsto \mathsf{a}, z \mapsto \mathsf{d} \,\} \subseteq \theta$ as $\neg\mathrm{p}(\mathsf{a},\mathsf{d})$ is the only negative literal in h'. In turn, $\mathrm{p}(x,y)\theta = \mathrm{p}(\mathsf{a},y)$ can be in $\mathsf{Lits}(h')$ only if $(y \mapsto \mathsf{b}) \in \theta$. But then, $\mathrm{p}(y,z)\theta = \mathrm{p}(\mathsf{b},\mathsf{d})$ is not in $\mathsf{Lits}(h')$.

2. $h' \models h$ is true. This is due to the soundness theorem and the fact that $h' \vdash h$. To prove the latter, we prove the equivalent relation $h' \wedge \neg h \vdash \Box$, i.e., iff the clause set

$$\{\, \neg h \,\} \cup \mathsf{Lits}(h') = \left\{ \begin{array}{l} \neg\mathrm{p}(x,y) \vee \neg\mathrm{p}(y,z) \vee \mathrm{p}(x,z) \\ \mathrm{p}(\mathsf{a},\mathsf{b}) \\ \mathrm{p}(\mathsf{b},\mathsf{c}) \\ \mathrm{p}(\mathsf{c},\mathsf{d}) \\ \neg\mathrm{p}(\mathsf{a},\mathsf{d}) \end{array} \right\}$$

is contradictory. We show the contradiction as follows

   (a) Resolve $\neg h\theta$, $\theta = \{\, x \mapsto \mathsf{a}, z \mapsto \mathsf{d} \,\}$ with $\neg\mathrm{p}(\mathsf{a},\mathsf{d})$ obtaining the resolvent clause

   $$c_1 = \neg\mathrm{p}(\mathsf{a},y) \vee \neg\mathrm{p}(y,\mathsf{d})$$

   (b) Resolve $c_1\theta$, $\theta = \{\, y \mapsto \mathsf{b} \,\}$ with $\mathrm{p}(\mathsf{a},\mathsf{b})$ obtaining

   $$c_2 = \neg\mathrm{p}(\mathsf{b},\mathsf{d})$$

   (c) Resolve $\neg h\theta$, $\theta = \{\, x \mapsto \mathsf{b}, z \mapsto \mathsf{d} \,\}$ with c_2 obtaining

   $$c_3 = \neg\mathrm{p}(\mathsf{b},y) \vee \neg\mathrm{p}(y,\mathsf{d})$$

   (d) Resolve $c_3\theta$, $\theta = \{\, y \mapsto \mathsf{c} \,\}$ with $\mathrm{p}(\mathsf{b},\mathsf{c})$ obtaining

   $$c_4 = \neg\mathrm{p}(\mathsf{c},\mathsf{d})$$

   (e) Finally, resolve c_4 with $\mathrm{p}(\mathsf{c},\mathsf{d})$ obtaining the empty clause $\Box$.

Note that there is a simple intuition behind the formal proof: $\neg h$ can be written also as $\mathrm{p}(x,y) \wedge \mathrm{p}(y,z) \rightarrow \mathrm{p}(x,z)$ expressing the transitivity property of $\mathrm{p}/2$. That property is in turn in obvious contradiction with all of $\mathrm{p}(\mathsf{a},\mathsf{b}), \mathrm{p}(\mathsf{b},\mathsf{c}), \mathrm{p}(\mathsf{c},\mathsf{d})$ being true while $\mathrm{p}(\mathsf{a},\mathsf{d})$ being false.

Question 15. (5 points)

Consider the following statements

1. X = non-self-resolving FOL clauses
2. X = contingent FOL clauses
3. There is no $k \in \mathbb{N}$, $x \in X$ such that $h_k \models x$ and $h_k \not\subseteq_\theta x$, where h_k $(k \in \mathbb{N})$ are the hypotheses of the generalization algorithm.

Decide for each of the implications $1 \rightarrow 2$, $1 \rightarrow 3$, $2 \rightarrow 3$, whether it is true. Change the relation $h_k \models x$ in (3) so that all the implications you decided true are true when (1) and (2) assume conjunctions instead of clauses.

Answer:

1. $1 \rightarrow 2$ is not true because an empty clause is not self-resolving but it is not contingent. The implication is however true for *non-empty* clauses. Assume for contradiction that a non-empty $x \in X$ is not self-resolving but not contingent, i.e. it is tautologically false or true. First assume the former case, i.e. $\models \neg x$. This means that for any formula x', it holds $x' \models \neg x$, i.e. $x \models \neg x'$. Since x is not self-resolving, this means $x \subseteq_\theta \neg x'$ for any clause $\neg x'$. This can only happen if $\mathsf{Lits}(x)$ is empty but then x is an empty clause, which contradicts the assumption. The case for $\models x$ is analogical (use the fact that the negation of a non-self-resolving clause is a non-self-resolving conjunction).

2. $1 \to 3$ is true. $h_k \models x$ and $h_k \not\subseteq_\theta x$ can happen simultaneously only if h_k is self-resolving and we will prove the implication by showing that for all $k \in \mathbb{N}$, h_k is not self-resolving.

   By the generalization algorithm, h_1 is the first positive example, which is from X so it is not self-resolving by the assumption of the implication. Assume for contradiction that for some $k \in \mathbb{N}$,

$$h_{k+1} = \mathsf{lgg}(h_k, x),\ x \in X \tag{24}$$

   is self-resolving, i.e., contains both a positive literal and a negative literal with the same predicate $\mathcal{P}$. By definition of the lgg operator, $\mathsf{lgg}(h_k, x)$ contains a predicate only if it occurs in both of h_k, x with the same sign. Thus x must also contain both a positive literal and a negative literal with the predicate $\mathcal{P}$. Thus x is self-resolving, which contradicts the assumption.

3. $2 \to 3$ is not true. Let the first positive example and thus also h_1 be $\mathrm{p}(x) \to \mathrm{p}(\mathsf{f}(x))$, i.e., $\neg\mathrm{p}(x) \vee \mathrm{p}(\mathsf{f}(x))$. Clearly, it is contingent: it is false e.g. with Herbrand interpretation $\{\, \mathrm{p}(\mathsf{a}) \,\}$ and true with Herbrand the (infinite) interpretation $\{\, \mathrm{p}(\mathsf{a}), \mathrm{p}(\mathsf{f}(\mathsf{a})), \mathrm{p}(\mathsf{f}(\mathsf{f}(\mathsf{a}))), \dots \}$. But it is also self-resolving because it contains the predicate p/1 in both a negative and positive literal. Thus it can happen that $h_1 \models x$ and $h_1 \not\subseteq_\theta x$. Indeed, this happens e.g. with the contingent observation $x = \mathrm{p}(x) \to \mathrm{p}(\mathsf{f}(\mathsf{f}(x)))$.

   However, the implication is true in the special case of *propositional* clauses because a propositional self-resolving clause is necessarily non-contingent.

To make $1 \to 3$ true for conjunctions, change $h_k \models x$ to $x \models h_k$ in item 2 of the question and in item 2 of the answer.

Question 16. (1 points)

Find two different least general generalizations of $\mathrm{p}(\mathsf{a})$ and $\mathrm{p}(\mathsf{b}) \vee \mathrm{p}(\mathsf{c})$, prove that they are indeed generalizations of the two clauses and prove that they are mutually θ-equivalent. Explain why two least general generalizations of the same pair of clauses or conjunctions must be θ-equivalent.

Answer:

$\mathrm{p}(x)$ and $\mathrm{p}(y) \vee \mathrm{p}(z)$

$\mathrm{p}(x)\theta \subseteq \mathrm{p}(y) \vee \mathrm{p}(z)$ for $\theta = \{\, x \mapsto y \,\}$

$(\mathrm{p}(y) \vee \mathrm{p}(z))\,\theta \subseteq \mathrm{p}(x)$ for $\theta = \{\, y \mapsto x, z \mapsto x \,\}$

So unlike in the propositional case, a least general generalization of FOL clauses (or conjunctions) is not unique!

Consider for contradiction two least general generalizations g, g' of the same pair of clauses or conjunctions such that $g \subset_\theta g'$. Then by the definition of least general generalization, g is not a least general generalization.

Question 17. (10 points)

Explain why the proof of the mistake bound n of the generalization algorithm is no longer valid when the assumption on X is changed to X = non-self-resolving FOL conjunctions or non-self-resolving FOL conjunctions clauses, the relations $\subseteq, \subset$ are changed to $\subseteq_\theta, \subset_\theta$ (respectively), and we set $n = |\mathcal{P}|$. Show that the proof cannot be rectified, in particular that no finite mistake bound exists under said assumption even if $\mathcal{F} = \emptyset$.

Answer:

The proof uses the following argument

> *Since examples are contingent, they have at most n literals (each of the n atoms is included either as a positive or negative literal but not both) and since h_1 is the first positive example, it also has at most n literals*

which is only true for propositional clauses or conjunctions. The number of different atoms using $n = |\mathcal{P}|$ predicates may be larger than n since there may be multiple atoms with the same predicate, each one using different terms as arguments.

Also the inference

> *at least one literal is removed on each mistake, so the* maximum number of mistakes is n

is not true in FOL. While a strict generalization ($h_{k+1} \subset_\theta h_k$) is still made after each mistake, this does not mean that $|\mathsf{Lits}(h_{k+1})| < |\mathsf{Lits}(h_k)|$ (consider e.g. $h_{k+1} = \mathrm{p}(x), h_k = \mathrm{p}(\mathsf{a})$.)

We can even show that no mistake bound can be provided under the assumption above. Let us assume for contradiction that the bound is M.

Let the target hypothesis be $\overline{h} = \mathrm{p}(z_1, z_2)$ and let the environment present the following positive examples x_k for $1 \le k \le M$

$$x_k = \bigvee_{1 \le i,j \le M-k+2, i \ne j} \mathrm{p}(z_i, z_j) \tag{25}$$

and

$$x_{M+1} = \mathrm{p}(z_1, z_2) \tag{26}$$

so e.g. for $M = 2$

$$\begin{aligned} x_1 &= \mathrm{p}(z_1,z_2) \vee \mathrm{p}(z_2,z_1) \vee \mathrm{p}(z_1,z_3) \vee \mathrm{p}(z_3,z_1) \vee \mathrm{p}(z_2,z_3) \vee \mathrm{p}(z_3,z_2) \\ x_2 &= \mathrm{p}(z_1,z_2) \vee \mathrm{p}(z_2,z_1) \\ x_3 &= \mathrm{p}(z_1,z_2) \end{aligned}$$

and since $x_3 \subset_\theta x_2 \subset_\theta x_1$ and the agent makes *least* generalizations, it will make $3 > M$ mistakes. In general, for any $M \in \mathbb{N}$, the sequence (25)-(26) is a chain of $M+1$ strict generalizations forcing the agent to make $M+1$ mistakes.

Note that for any $M \in \mathbb{N}$ and $k \in [1; M+1]$, $x_k \subseteq_\theta \mathrm{p}(z,z)$. Therefore, there is an infinite *chain of generalizations between two* finite *elements $\mathrm{p}(z_1,z_2)$ and $\mathrm{p}(z,z)$ in the θ-subsumption lattice!*

Question 18. (3 points)

Determine the least general generalization of the following two assertions

1. *Superman is mortal or he is not a human.*
2. *Every human who smokes is mortal.*

by representing them as first-order logic clauses and computing their least general generalization with respect to the θ-subsumption order, and express the result in natural language.

Answer:

The first clause is

$$\mathrm{mortal}(\mathsf{superman}) \vee \neg\mathrm{human}(\mathsf{superman})$$

The second clause is

$$\mathrm{human}(x) \wedge \mathrm{smokes}(x) \to \mathrm{mortal}(x)$$

which can be written as

$$\neg\mathrm{human}(x) \vee \neg\mathrm{smokes}(x) \vee \mathrm{mortal}(x)$$

The LGG is

$$\neg\mathrm{human}(\mathsf{lgg}(\mathsf{superman}, x)) \vee \mathrm{mortal}(\mathsf{lgg}(\mathsf{superman}, x)) \tag{27}$$

$$\text{i.e.,} \tag{28}$$

$$\neg\mathrm{human}(v_1) \vee \mathrm{mortal}(v_1) \tag{29}$$

$$\text{i.e.,} \tag{30}$$

$$\mathrm{human}(v_1) \to \mathrm{mortal}(v_1) \tag{31}$$

i.e., *every human is mortal.*

Question 19. (2 points)

Let h, h' be FOL clauses and B a ground FOL conjunction. Show that if $h \subseteq_\theta h'$ then $h \subseteq_\theta^B h'$.

Answer:

$h \subseteq_\theta h'$ means that

$$\exists\theta : \mathsf{Lits}(h\theta) \subseteq \mathsf{Lits}(h') \tag{32}$$

$h \subseteq_\theta^B h'$ is defined as $h \subseteq_\theta B \to h'$, i.e.,

$$\begin{aligned}
&\exists\theta : \mathsf{Lits}(h\theta) \subseteq \mathsf{Lits}(B \to h') \\
&\exists\theta : \mathsf{Lits}(h\theta) \subseteq \mathsf{Lits}(\neg B \vee h') \\
&\exists\theta : \mathsf{Lits}(h\theta) \subseteq \mathsf{Lits}(\neg B) \cup \mathsf{Lits}(h') \qquad (33)
\end{aligned}$$

where (33) is clearly implied by (32).

Question 20. (5 points)

Show that

$$h = \text{parent}(v_2, v_1) \wedge \text{male}(v_1) \to \text{son}(v_1, v_2)$$

and

$$\begin{aligned}
g = \text{son}(v_1, v_2) \vee \neg\text{female}(\mathsf{a}) \vee \neg\text{parent}(\mathsf{a}, \mathsf{b}) \vee \neg\text{parent}(v_2, v_1) \vee \neg\text{male}(\mathsf{b}) \vee \\
\neg\text{male}(v_1) \vee \neg\text{parent}(v_3, v_4) \vee \neg\text{parent}(\mathsf{b}, \mathsf{c}) \vee \neg\text{male}(v_4) \vee \neg\text{male}(\mathsf{c})
\end{aligned}$$

are equivalent relative to

$$B = \text{female}(\mathsf{a}) \wedge \text{parent}(\mathsf{a}, \mathsf{b}) \wedge \text{male}(\mathsf{b}) \wedge \text{parent}(\mathsf{b}, \mathsf{c}) \wedge \text{male}(\mathsf{c})$$

Answer:

$h \approx_\theta^B g$ iff $h \subseteq_\theta^B g$ and $g \subseteq_\theta^B h$.

$h \subseteq_\theta^B g$ because $h \subseteq_\theta g$, which is because $\mathsf{Lits}(h\theta) \subseteq \mathsf{Lits}(g)$ with $\theta = \emptyset$.

$g \subseteq_\theta^B h$ because $g \subseteq_\theta B \to h$, which is because $\mathsf{Lits}((B \to g)\theta) \subseteq \mathsf{Lits}(h)$ with $\theta = \{ v_3 \mapsto v_2, v_4 \mapsto v_1 \}$.

Question 21. (10 points)

Let

$$\begin{aligned}
B =& \text{half}(4, 2) \wedge \text{half}(2, 1) \wedge \text{int}(2) \wedge \text{int}(1) \\
x_1 =& \text{even}(4) \\
x_2 =& \text{even}(2)
\end{aligned}$$

1. Compute a least general generalization of x_1, x_2 observations relative to B .
2. Determine the reduction of the resulting clause relative to B and justify why it is indeed a reduction of it relative to B.

Answer:

1. $\text{rlgg}_B(x_1, x_2) = \mathsf{lgg}(B \to x_1, B \to x_2)$ where

$$B \to x_1 = \text{even}(4) \vee \neg\text{half}(4,2) \vee \neg\text{half}(2,1) \vee \neg\text{int}(2) \vee \neg\text{int}(1)$$
$$B \to x_2 = \text{even}(2) \vee \neg\text{half}(4,2) \vee \neg\text{half}(2,1) \vee \neg\text{int}(2) \vee \neg\text{int}(1)$$

	$\text{even}(4)$	$\neg\text{half}(4,2)$	$\neg\text{half}(2,1)$	$\neg\text{int}(2)$	$\neg\text{int}(1)$
$\text{even}(2)$	$\text{even}(v_1)$				
$\neg\text{half}(4,2)$		$\neg\text{half}(4,2)$	$\neg\text{half}(v_3,v_4)$		
$\neg\text{half}(2,1)$		$\neg\text{half}(v_1,v_2)$	$\neg\text{half}(2,1)$		
$\neg\text{int}(2)$				$\neg\text{int}(2)$	$\neg\text{int}(v_4)$
$\neg\text{int}(1)$				$\neg\text{int}(v_2)$	$\neg\text{int}(1)$

θ	σ	new variable
2	4	v_1
1	2	v_2
4	2	v_3
2	1	v_4

$$\mathsf{lgg} = \text{even}(v_1) \vee \neg\text{half}(4,2) \vee \neg\text{half}(v_3,v_4) \vee \neg\text{half}(v_1,v_2) \vee \neg\text{half}(2,1) \vee \neg\text{int}(2) \vee \neg\text{int}(v_4) \vee \neg\text{int}(v_2) \vee \neg\text{int}(1)$$

or

$$\text{even}(v_1) \leftarrow \text{half}(4,2) \wedge \text{half}(v_3,v_4) \wedge \text{half}(v_1,v_2) \wedge \text{half}(2,1) \wedge \text{int}(2) \wedge \text{int}(v_4) \wedge \text{int}(v_2) \wedge \text{int}(1)$$

2. Remove ground literals present in B obtaining a clause which is subsume-equivalent (relative to B) to the lgg.

$$\text{even}(v_1) \leftarrow \text{half}(v_3,v_4) \wedge \text{half}(v_1,v_2) \wedge \text{int}(v_4) \wedge \text{int}(v_2)$$

This is equivalent to

$$\text{even}(v_1) \leftarrow \text{half}(v_1,v_2) \wedge \text{int}(v_2)$$

The latter subsumes the former evidently as its is a subset of the former's literals. The former subsumes the latter under substitution $\theta = \{ v_3 \mapsto v_1, v_4 \mapsto v_2 \}$. The last clause is a reduction of the LGG as it is subsume-equivalent to it relative to B as shown above, and it cannot be reduced further.

Question 22. (10 points)

Let X contain Herbrand interpretations for a finite set of $\mathcal{P}$ predicates and a finite set $\mathcal{F}$ of functions, and the observation complexity n_X be the tuple $(|\mathcal{P}|, |\mathcal{F}|)$. Show that the hypothesis class *st*-CNF (i.e., conjunctions of FOL clauses with at most s literals and at most t term occurrences in each literal) is learnable online from X.

Answer:

We reduce online *st*-CNF learning from X to learning a propositional monotone conjunction from truth assignments. More precisely, let $c_1, c_2, \ldots, c_{n'}$ be all *st* clauses made using $\mathcal{P}$ and $\mathcal{F}$. We learn a monotone propositional conjunction on n' variables from $\{0,1\}^{n'}$; for each observation $x \in X$ we present the learner examples $x'(x) \in \{0,1\}^{n'}$ such that

$$x'^i(x) = 1 \text{ iff } x \models c_i$$

We know that the monotone propositional conjunction can be learned online from $\{0,1\}^{n'}$ with mistake bound $poly(n')$. To show online learnability *st*-CNF from X, we only need to show $n' \leq poly(n_X) = poly(|\mathcal{P}|, |\mathcal{F}|)$.

An *st*-clause has no more that st different variables. So the maximum number of different terms in an *st*-clause is $|\mathcal{F}| + st$. An atom consists of a single predicate ($|\mathcal{P}|$ different choices) and at most t terms (each from from $|\mathcal{F}| + st$ choices). So there are $|\mathcal{P}|(|\mathcal{F}| + st)^t$ different atoms, i.e. $2|\mathcal{P}|(|\mathcal{F}| + st)^t$ different literals.

An *st*-clause combines at most s literals so there are at most

$$\mathcal{O}\binom{2|\mathcal{P}|(|\mathcal{F}| + st)^t}{s} = poly(|\mathcal{P}|, |\mathcal{F}|)$$

different *st*-clauses.

Question 23. (2 points)

Consider a version-space agent whose initial hypothesis class $\mathcal{H}_1$ contains all non-contradictory conjunctions on 3 propositional variables.

1. Determine $|\mathcal{H}_1|$.

   Answer:

   $3^3 = 27$ (Each of the 3 variables may be absent, positive, or negative in the conjunction.)

2. Give an upper bound on $|\mathcal{H}_2|$ given that $r_2 = -1$.

   Answer:

   A version-space agent decides by majority vote so at least 14 hypotheses in $\mathcal{H}$ were inconsistent with x_1; these get deleted and at most 13 remain.

Question 24. (2 points)

Let $r_{\leq K}$ be a reward sequence of a standard agent and $h_{\leq K}$ be its sequence of hypotheses. Denote $M = \sum_{k=1}^{K} |r_k|$. Show that there is a hypothesis h retained for at least $\frac{K}{M+1}$ consecutive steps in $h_{\leq K}$, i.e.

$$h_{\leq K} = h_1, h_2, \ldots \underbrace{h, h, \ldots h}_{\text{at least } \frac{K}{M+1} \text{ times}}, \ldots h_K$$

Answer:

A standard agent changes its hypothesis ($h_{k+1} \neq h_k$) if and only if a mistake is made ($r_k = -1$). The number of mistakes up to time K is M so there are at most $M+1$ interleaving subsequences of unchanged hypotheses in $h_{\leq K}$ (there are exactly M of them if the M's mistake is made at time K, otherwise $M+1$). If each of the at most $M+1$ subsequences is shorter than $\frac{K}{M+1}$ then the length of the sequence is K less than $\frac{K}{M+1} \cdot (M+1) = K$, which is a contradiction.

Question 25. (5 points)

Let an agent PAC-learn $\mathcal{C}$ from X. Show that for any target concept from $\mathcal{C}$ on X, an arbitrary distribution $P(x)$ on X and arbitrary numbers $0 < \epsilon, \delta < 1$ and $K \in \mathbb{N}$, the condition $\mathsf{err}(h_K) \leq \epsilon$ with probability at least $1 - \delta$ implies that h_K is consistent with all observations in $x_{<K}$.

Answer:

Assume for contradiction that h_K is not consistent with some x_k, $1 \leq k \leq K$. The distribution $P(x)$ and the number $\delta < 1$ are arbitrary, so set them so that $\prod_{i=1}^{K} P(x_i) > \delta$ (which can evidently be done). This implies that $P(x_k) > 0$. The number $\epsilon > 0$ is also arbitrary so set it so that $\epsilon < P(x_k)$. Since h_K is inconsistent with x_k, $\mathsf{err}(h_K) \geq P(x_k) > \epsilon$. This happens with probability $\prod_{i=1}^{K} P(x_i) > \delta$ which contradicts the assumption of the implication.

Question 26. (5 points)

Let X contain all real numbers from $[0; 1]$ which can be represented using 256 bits. Let $\mathcal{H} = X$, and the decision policy given by a $h \in \mathcal{H}$ is

$$h(x) = 1 \text{ iff } x > h$$

Determine a k such that with probability at least 0.9, $\mathsf{err}(h) < 0.1$, where h is an arbitrary hypothesis from $\mathcal{H}$ consistent with k i.i.d. examples from X. Estimate it using:

1. $\ln |\mathcal{H}|$
2. $\mathsf{VC}(\mathcal{H})$

Answer:

1. The probability that some of the 2^{256} hypotheses is bad ($\mathsf{err}(h) > 0.1$) and still consistent with k i.i.d. observations is at most $2^{256}(1 - 0.1)^k = 2^{256}(0.9)^k$. We want this probability to be smaller than $1 - 0.9 = 0.1$:

$$2^{256}(0.9)^k < 0.1$$

i.e.

$$k > \log_{0.9} \frac{0.1}{2^{256}} \approx 1707 \text{ examples (smalllest such } k) \tag{34}$$

We can also use the general formula (see Lectures)

$$k > \frac{1}{\epsilon} \ln \frac{|\mathcal{H}|}{\delta} = \frac{1}{0.1} \ln \frac{2^{256}}{0.1} \approx 1798 \text{ examples (smalllest such } k)$$

which is slightly larger than (34) because the upper bound $(1-\epsilon)^k < e^{-\epsilon k} (\epsilon > 0)$ is used in the derivation of the formula.

2. Using $\mathsf{VC}(\mathcal{H})$. $\mathsf{VC}(\mathcal{H}) = 1$ because a single number from X can evidently be shattered (classified positively or negatively by hypotheses from $\mathcal{H}$) but two different numbers from X cannot be shattered: the smaller cannot be made positive while the larger is negative.

$$k > \max\left\{ \frac{4}{\epsilon} \lg \frac{2}{\delta}, \frac{8 \cdot \mathsf{VC}(\mathcal{H})}{\epsilon} \lg \frac{13}{\epsilon} \right\} \approx \max\{\, 173, 562 \,\} = 562 \text{ examples}$$

Question 27. (3 points)

Give a real-life meaning to binary random variables A, B, C, for which the following Bayes graph is appropriate

1. $$\begin{tikzcd} \textcircled{A} \arrow[r] & \textcircled{B} \arrow[r] & \textcircled{C} \end{tikzcd}$$
2. $$\begin{tikzcd} \textcircled{A} \arrow[r] & \textcircled{B} & \textcircled{C} \arrow[l] \end{tikzcd}$$
3. $$\begin{tikzcd} \textcircled{A} & \textcircled{B} \arrow[l] \arrow[r] & \textcircled{C} \end{tikzcd}$$

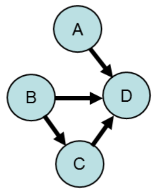
For each case, decide if the graph implies $A \perp\!\!\!\perp_P C \mid B$.

Answer:

1. A : thorough studying, B : good understanding, C : successful exam. YES
2. A : flu, B : fever, C : pneumonia. NO
3. A : high cancer incidence, B : nuclear blast, C : compromised environment. YES

Question 28. (2 points)

How many parameters are needed in the Bayesian network below to fully specify a joint distribution on the random variables in vertices, which are binary?

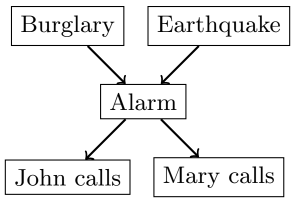


Answer:

Summing parameters in vertices in alphabetica order: $2^0 + 2^0 + 2^1 + 2^3 = 12$

Question 29. (5 points)

Given the Bayesian network below (conditional probability tables not shown),



1. (1 point) list all variables that 'John calls' is independent of, given that 'Alarm' is observed.

   Answer:

   Burglary, Earthquake, Mary calls

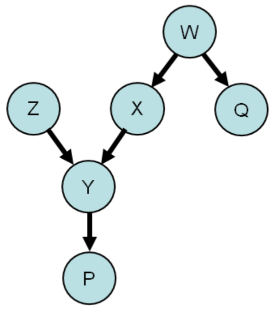
2. (4 points) express the probability that neither Mary nor John calls given that both burglary and earthquake happens, using only those (conditional) probabilities which are encoded in the conditional probability tables appropriate for this network. (Produce a formula referring to the random events by symbols such as A and their outcomes by symbols such as $b, \neg m$, etc.).

   Answer:

$$\begin{aligned}\frac{P(\neg m, \neg j, b, e)}{P(b,e)} &= \\ \frac{\sum_A P(b)P(e)P(A|b,e)P(\neg j|A)P(\neg m|A)}{P(b)P(e)} &= \\ \sum_A P(A|b,e)P(\neg j|A)P(\neg m|A).\end{aligned}$$

Question 30. (5 points)

Let $X_1, X_2, \ldots \perp\!\!\!\perp_P Y_1, Y_2, \ldots \mid \mathcal{E}$ denote that $\forall i,j \in \{1,2,\ldots\} : X_i \perp\!\!\!\perp_P Y_j \mid \mathcal{E}$. The empty set is denoted as $\emptyset$. Decide (true/false) for each of the statements below whether it is implied by the Bayes graph for P:



1. $Q \perp\!\!\!\perp_P X, Y, Z, P \mid W$,

   Answer:

   True, Q ← W → X, diverging connection, W observed

2. $Z, Y, P \perp\!\!\!\perp_P W, Q \mid \emptyset$,

   Answer:

   False, W → X → Y linear, X not observed

3. $Z \perp\!\!\!\perp_P X, W, Q \mid \emptyset$,

   Answer:

   True Z → Y ← X converging connection, neither Y nor P observed

4. $Z \perp\!\!\!\perp_P X, W, Q \mid P$,

   Answer:

   False, Z $\rightarrow$ Y $\leftarrow$ X converging connection, descendant P observed

5. $Z, Y, P \perp\!\!\!\perp_P W, Q \mid X$,

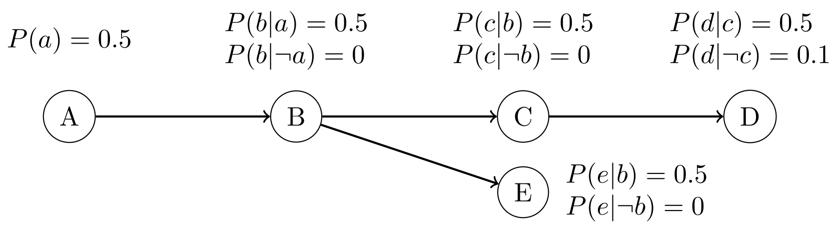
   Answer:

   True, W $\rightarrow$ X $\rightarrow$ Y linear connection, X observed

---

Question 31. (15 points)

Consider the Bayes Network



1. Calculate $P(a|d)$ and $P(\neg a|d)$ by the factor method, i.e., using factor multiplication and variable elimination.
2. Determine $\arg\max_{A,B} P(A, B|d)$ by the factor method.

Answer:

1. E can be ignored as it is a terminal vertex which is not part of the query or evidence.

$$P(A|d) = \alpha \sum_{B,C} P(A)P(B|A)P(C|B)P(d|C)$$

Push sums to the right

$$= \alpha P(A) \sum_B P(B|A) \sum_C P(C|B)P(d|C)$$

Reformulate with factors

$$= \alpha \begin{bmatrix} 0.5 & \neg a \\ 0.5 & a \end{bmatrix} \times \sum_B \begin{bmatrix} 1 & \neg a, \neg b \\ 0 & \neg a, b \\ 0.5 & a, \neg b \\ 0.5 & a, b \end{bmatrix} \times \sum_C \begin{bmatrix} 1 & \neg b, \neg c \\ 0 & \neg b, c \\ 0.5 & b, \neg c \\ 0.5 & b, c \end{bmatrix} \times \begin{bmatrix} 0.1 & \neg c \\ 0.5 & c \end{bmatrix}$$

Multiply last two factors

$$= \alpha \begin{bmatrix} 0.5 & \neg a \\ 0.5 & a \end{bmatrix} \times \sum_B \begin{bmatrix} 1 & \neg a, \neg b \\ 0 & \neg a, b \\ 0.5 & a, \neg b \\ 0.5 & a, b \end{bmatrix} \times \sum_C \begin{bmatrix} 0.1 & \neg b, \neg c \\ 0 & \neg b, c \\ 0.05 & b, \neg c \\ 0.25 & b, c \end{bmatrix}$$

Eliminate C:

$$= \alpha \begin{bmatrix} 0.5 & \neg a \\ 0.5 & a \end{bmatrix} \times \sum_B \begin{bmatrix} 1 & \neg a, \neg b \\ 0 & \neg a, b \\ 0.5 & a, \neg b \\ 0.5 & a, b \end{bmatrix} \times \begin{bmatrix} 0.1 & \neg b \\ 0.3 & b \end{bmatrix}$$

Multiply last two factors

$$= \alpha \begin{bmatrix} 0.5 & \neg a \\ 0.5 & a \end{bmatrix} \times \sum_B \begin{bmatrix} 0.1 & \neg a, \neg b \\ 0 & \neg a, b \\ 0.05 & a, \neg b \\ 0.15 & a, b \end{bmatrix}$$

Eliminate B:

$$= \alpha \begin{bmatrix} 0.5 & \neg a \\ 0.5 & a \end{bmatrix} \times \begin{bmatrix} 0.1 & \neg a \\ 0.2 & a \end{bmatrix}$$

$$= \alpha \underbrace{\begin{bmatrix} 0.05 & \neg a \\ 0.1 & a \end{bmatrix}}_{=f(A)}$$

$P(a) + P(\neg a) = \alpha f(a) + \alpha f(\neg a) = 1$. Since $\alpha f(a) = 2\alpha f(\neg a)$, we have $P(a) = 2/3$ and $P(\neg a) = 1/3$.

2. Reuse the computation above up until the elimination of B (since B will be maximized-out instead):

$$\max_{A,B} P(A,B|d) = \alpha \max_A \begin{bmatrix} 0.5 & \neg a \\ 0.5 & a \end{bmatrix} \times \max_B \underbrace{\begin{bmatrix} 0.1 & \neg a, \neg b \\ 0 & \neg a, b \\ 0.05 & a, \neg b \\ 0.15 & a, b \end{bmatrix}}_{=f_1(A,B)}$$

Maximize-out B :

$$= \alpha \max_A \begin{bmatrix} 0.5 & \neg a \\ 0.5 & a \end{bmatrix} \times \begin{bmatrix} 0.1 & \neg a \\ 0.15 & a \end{bmatrix}$$

$$= \alpha \max_A \underbrace{\begin{bmatrix} 0.05 & \neg a \\ 0.075 & a \end{bmatrix}}_{f_2(A)}$$

$\arg\max_A f_2(A) = a$

$\arg\max_B f_1(a, B) = b$

So $\arg\max_{A,B} P(A,B|d) = (a, b)$

Question 32. (8 points)

Consider a Markov decision process.

1. (3 points) Explain why a fixed (independent of $x_1, x_2, \ldots$) sequence of actions as $y_1, y_2, \ldots$ ($y_k \in Y$) does not solve a Markov decision process, i.e. cannot guarantee optimality in reinforcement learning. In which field of AI would a fixed sequence of actions be an appropriate solution? Where is the boundary between game theory and reinforcement learning?

   Answer:

   No fixed sequence of actions is guaranteed to be optimal in MDPs as the environment is stochastic and the current action must be chosen in dependence to the observed state. A fixed sequence of states is a valid solution for classical (i.e., not probabilistic) planning. The MDPs and RL setting can be seen, therefore, as a generalization of the planning. The game theory field is generalization of the MDPs and RL as it includes possibility for multiple agents and not only stochastic environment.

2. (2 points) Recall the value iteration algorithm. This algorithm is based on a general method for solving a set of equations. Give the name for this method and explain how it works in one or two sequences.

   Answer:

   The value iteration algorithm is based on fixed point method (in Czech, *metoda prosté iterace*) for finding a solution of a set of equations. This method is based on iterative application of the Bellman operator on the initial estimate of the utility.

3. (3 points) Recall the policy iteration algorithm. This algorithm is based on another well known concept from machine learning and statistics. Name this algorithm and explain its idea in one or two sequences. Provide an example of other usages of this algorithm in computer science or mathematics.

   Answer:

   The policy iteration algorithm is based on the same idea as the expectation–maximization (EM) algorithm. In this algorithm, we aim on calculating the estimates of the utility $\widehat{U}$, having the max operator in the set of equations. Instead of solving this set of equations directly, we define a set of additional parameters (the policy). Then we iteratively optimize the utility estimates with respect to a fixed policy, followed by a step where we optimize the policy w.r.t. fixed utility estimates. Alternating those two steps is guaranteed to find an optimal policy. The EM algorithm can be found (among others) in Baum-Welsch algorithm for learning hidden Marcov models or estimation of the parameters for a mixture of Gaussians.

Question 33. (5 points)

We state the Bellman equations the following way:

$$U(x) = r(x) + \gamma \max_{y \in Y(x)} \sum_{x'} P(x' \mid x, y) U(x').$$

In some literature, you may find under the same name a different equation:

$$U(x) = \max_{y \in Y(x)} \sum_{x'} P(x' \mid x, y)(r(x, y, x') + \gamma U(x'))$$

Describe in natural language the difference between the two formulations and decide if they are equally general, or else describe which one is the more general and why.

Answer:

In the first representation, the reward depends only on the current state, in the second the reward depends on the previous state, action taken, and the current state. Assuming that the second two parameters of the reward are irrelevant, i.e.,

$r(x, y, x') = r(x)$, we can rewrite the second representation as

$$\begin{aligned}
&\max_{y \in Y(x)} \sum_{x'} P(x' \mid x, y)(r(x) + \gamma U(x')) = \\
&\max_{y \in Y(x)} \left(\sum_{x'} P(x' \mid x, y) r(x) + \gamma \sum_{x'} P(x' \mid x, y) U(x') \right) = \\
&\max_{y \in Y(x)} \left(r(x) + \gamma \sum_{x'} P(x' \mid x, y) U(x') \right) = \\
&r(x) + \gamma \max_{y \in Y(x)} \sum_{x'} P(x' \mid x, y) U(x'),
\end{aligned}$$

which shows that the second representation is as least as general as the first one. It remains to show that the first representation can model anything that the second one can. The key idea is to modify the state space. We can create a new state for each possible combination of state-action-state by defining $x_{\mathsf{new}} = (x, y, x')$. This inflation of the state space allows us to make reward dependent only on the state. (By the way, the modification is the same if we wanted to translate MDPs of the second-order to MPDs of the first order.) Therefore, both situations are equivalent, and we can deal only with the first one.

Question 34. (12 points)

Despite many reinforcement learning algorithms with additive rewards, it is common to use discounted rewards to model the environment.

1. (1 point) Give the range for the discount factor parameter.

   Answer:

   The discount factor is from $[0, 1]$.

2. (1 point) How does the agent behave when $\gamma = 0$?

   Answer:

   The agent behaves greedily, goes only for the highest reward in the next step.

3. (2 points) Give an example of an environment where a high discount factor is a good choice and why. Do the same for a low discount factor.

   Answer:

   The high discount factor is appropriate when it is irrelevant how long it will take the agent to obtain the reward in a terminal state. For example, when playing chess, it is more relevant whether the agent wins or loses than the number of moves. Therefore, the discount factor will be close to one. If our goal is to get the highest number of wins, then the discount factor will be exactly one.

   A small discount factor might be used in scenarios when the agent is in a very unstable environment. Imagine a robot on Mars that wants to decide whether to send experimental results to Earth. As dust storms are likely, it is wise to send data earlier, meaning low discount factor. Similar results can be achieved by a small negative reward in each non-terminal state.

4. (3 points) In the case of the infinite horizon, discounting the rewards is necessary. Explain why.

   Answer:

   If some of the agent histories are infinite with repeated positive rewards, it may happen that infinite sequences are not comparable as, for example, $1 + 1 + 1 + \cdots$ and $2 + 2 + 2 + \cdots$ have both values ∞. The second one is, however, clearly preferred. Discounting allows us to distinguish between those two as for any $\gamma \in (0, 1)$, the second one has a higher discounted sum.

5. (3 points) Imagine that your fancy reinforcement learning algorithm is not working on a chess game. Is it a good idea to include the discount factor in grid-search on meta-parameters of your algorithm? If yes, explain why; if not, would you still consider a range of discount parameter values and why?

   Answer:

   The discount factor is part of the environment. In the chess game, the reward is only dependent on whether we won or lost. The rule of repeating a position three times means that the game has a finite horizon. Hence, the discount factor should be one. Grid-search on the value of discount factor might sometimes give good results for γ close to zero. This, however, in the case of chess, indicates either an ill-working algorithm or too short learning (i.e., too few learning episodes) as it is easier to learn when you limit the horizon. In an extreme case, when $\gamma = 0$, no learning is needed at all. In the case of chess, we might still consider values close to one but not one exactly (i.e., 0.99 or 0.999) as it improves the numerical stability of some algorithms by distinguishing paths to terminal states of different lengths.

In the case of finite-horizon game, when winning/loose situations are only necessary, it can be mathematically proven (try it yourself) that value of a state is equal to the expected number of wins weighted by the terminal state rewards.

There are still some scenarios when grid search on discount factor is needed, especially when the discount factor is unknown and cannot be estimated. However, caution is needed.

6. (2 points) Suppose that

$$\max_{x \in X} |r(x)| = r_{\mathsf{max}}.$$

Using only r_{max} and γ, give the tightest lower and upper bound on the cumulative discounted reward in a single episode.

Answer:

The maximum is equal to

$$r_{\mathsf{max}} + \gamma \cdot r_{\mathsf{max}} + \gamma^2 \cdot r_{\mathsf{max}} + \cdots = \frac{1}{1-\gamma} r_{\mathsf{max}}.$$

The minimum is the same except with the minus sign.

Question 35. (5 points)

Recall the learning rate parameter α of the temporal difference learning.

1. (1 point) Provide range for the α parameter. **Answer:**

   It must always hold that $\alpha \in (0, 1)$.

2. (1 point) Explain the meaning of the α parameter. **Answer:**

   The learning rate α is a meta-parameter of the learning algorithm. It regulates the learning speed by balancing the newly seen sample and previous samples stored in the weights of the model.

3. (4 points) What must hold for α so that the temporal difference learning converges? **Answer:**

   The following must be true:

$$\sum_{k=0}^{\infty} \alpha_k = \infty, \qquad \sum_{k=0}^{\infty} \alpha_k^2 < \infty,$$

   i.e., the first sum must diverge, the second one must converge.

   The previous formula is a more general concept and does not hold only in TD-learning. See the next question.

4. (1 point) Relate the temporal difference update rule

$$\widehat{U}(x) := \widehat{U}(x) + \alpha \left(r(x) + \gamma \cdot \widehat{U}(x') - \widehat{U}(x) \right)$$

   to another well-known algorithm used in mathematical optimization. **Answer:**

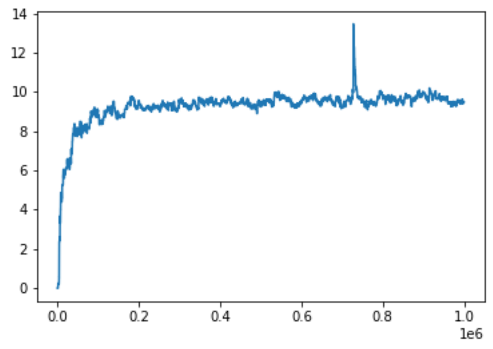
   The temporal difference learning rule is an instance of gradient descent.

   This TD-update is a special case of the Widrow-Hoff rule.

Question 36. (13 points)

In this problem, we will study the influence of learning rate α on the value estimates $\widehat{U}$. All figures show learning of U using the temporal-difference method for the same state over one million episodes. The learning rate was selected so that the conditions for convergence were met.

1. (2 points) Explain what causes the spike that you see around episode 700000.



Answer:

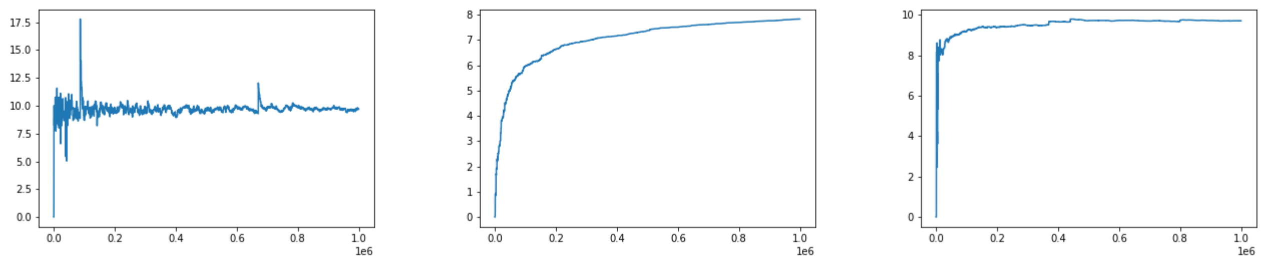
Learning is a stochastic process in a stochastic environment. Therefore, a rare event leading to a high reward might cause such a peak in learning.

2. (2 points) Why do we need a different learning rate value for each state.

Answer:

The learning rate is a function of the number of visits of a state. For example, $\alpha(n_x) = \frac{1}{n_x}$. However, not all states are visited with the same frequency. Therefore, to speed up the learning process, we set different learning rates for different states. States that are visited frequently can converge faster than states that are visited rarely.

3. (6 points) Consider the following three scenarios of learning the value of a single state under a different learning rate. Explain which situation you consider optimal and identify when the learning rate was too small or too big. Propose a solution for the suboptimal cases.



Answer:

The first figure is an example of learning with a too high learning rate. This is explained by the shattering and high peaks in the beginning. As a result, the value estimates are oscillating around $U(x)$ even with one million epochs. The solution is either a lower learning rate or a higher number of learning episodes.

The second figure is an example of a too low learning rate. The values did not converge and are still rising to the $U(x)$. The solution is either to use a higher number of learning episodes, or to start with higher learning rate values.

The third figure does not suffer any of the problems stated above and should be considered a good example for learning of the state presented.

4. (3 points) The learning rate is a function of number of visits of a state $\alpha(n_x)$. Consider the following three functions

$$\alpha_1(n_x) = \frac{1}{10 + n_x}, \qquad \alpha_2(n_x) = \frac{3}{2 + n_x}, \qquad \alpha_3(n_x) = \frac{100}{99 + n_x}.$$

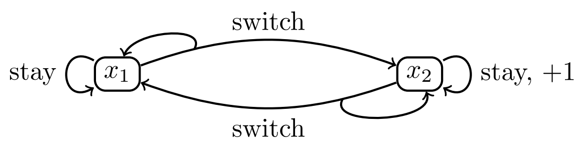
The figures in the question 3 were generated using those three learning rate functions. Match those functions to the figures and explain your decision.

Answer:

Learning rate α_1 is the smallest, α_3 is the highest. Therefore the figures were generated using α_3, α_1, and α_2 respectively.

Question 37. (10 points)

Consider the following MDP. Assume that reward is in the form $r(x, y)$, i.e., $r : X \times Y \mapsto \mathbb{R}$. Set $\gamma = \frac{1}{2}$.



Suppose that you have seen the following sequence of states, actions, and rewards:

$$x_1, \text{switch}, x_2, \text{stay}, +1, x_2, \text{stay}, +1, x_2, \text{switch}, x_1, \text{stay}, x_1, \text{switch}, x_1, \text{switch}, x_1, \text{stay}, x_1, \text{switch}, x_2, \text{stay}, +1, x_2$$

1. (4 points) What is $\widehat{U}^\pi(x_i)$ calculated by the Direct Utility Estimation algorithm?
2. (2 points) What is transition model P estimated by the Adaptive Dynamic Programming algorithm?
3. (2 points) In the ADP estimates, some of the rare events might have zero probability, even though they are possible. Provide a solution in which the rare events that the algorithm misses during learning have a non-zero probability.
4. (2 points) What are state values estimated by a Temporal Difference learning agent after two steps? Assume that $\alpha = 0.1$ and all values are initialized to zero.

   Answer:

   After the first interaction, the update to $\widehat{U}^\pi(x_1)$ is made so that

   $$\widehat{U}^\pi_1(x_1) = \widehat{U}^\pi_0(x_1) + \alpha(r(x_1, \text{switch}) + \gamma \widehat{U}^\pi_0(x_2) - \widehat{U}^\pi_0(x_1)) = 0 + 0.1(0 + 0.5 \cdot 0 - 0) = 0.$$

   No update is made to $\widehat{U}^\pi(x_2)$, i.e., $\widehat{U}^\pi_0(x_2) = \widehat{U}^\pi_1(x_2) = 0$.

   After the second interaction, no update is made to $\widehat{U}^\pi(x_1)$, however,

   $$\widehat{U}^\pi_2(x_2) = \widehat{U}^\pi_1(x_2) + \alpha(r(x_2, \text{switch}) + \gamma \widehat{U}^\pi_1(x_2) - \widehat{U}^\pi_1(x_2)) = 0 + 0.1(1 + 0.5 \cdot 0 - 0) = 0.1.$$

   Answer:

   $Q(x, y)$ is very likely between 9 and 10, as figures 1 and 4 indicate. Figures 2 and 3 show insufficient learning. Figures 1 and 4 agree on this number.

[adapted from Richard Sutton's 609 course, see `http://www.incompleteideas.net/book/the-book-2nd.html`]

Question 38. (3 points)

Decide whether the following statement is true or false: *If a policy π is greedy with respect to its own value function U^π, then this policy is an optimal policy.* Explain your decision.

[adapted from Richard Sutton's 609 course, see `http://www.incompleteideas.net/book/the-book-2nd.html`]

Answer:

The statement is true. We can argue by the policy iteration algorithm, which is guaranteed to find an optimal policy. The condition needed for the algorithm to end is that the policy does not change in two consecutive steps - evaluation of a policy and successive calculation of the greedy policy. In other words, the optimal policy found by this algorithm is greedy with respect to its value function.

However, the reasoning is still not complete as the policy found by the policy iteration algorithm might differ from policy π. This discrepancy can be easily fixed by initializing the policy iteration algorithm with π. The algorithm terminates immediately, meaning that the policy is optimal.

Question 39. (5 points)

Do the following exploration/exploitation schemes fulfill the 'infinite exploration' and 'greedy in limit' conditions? Which lead to the convergence of Q-values in Q-learning and which lead to the convergence of Q-values in SARSA. Does anything change if we are interested in the convergence of policy? $n_{x,y}$ denotes the number of times when action y was taken in state x. n_y is defined similarly.

1. a random policy
2. $$\pi(x) = \begin{cases} y, & \text{if } n_{x,y} \leq 100, \\ \arg\max_y Q(x,y), & \text{otherwise.} \end{cases}$$
3. ε-greedy policy with $\varepsilon = \frac{1}{n_x^2}$
4. ε-greedy policy with $\varepsilon = \frac{1{,}000}{999+n_x}$
5. ε-greedy policy with $\varepsilon = \frac{1}{\sqrt{n_x}}$

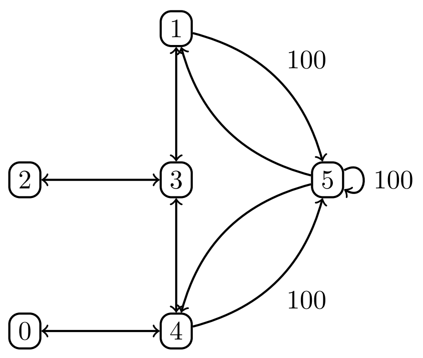
Answer:

Convergence of Q-values in Q-learning is guaranteed when the 'infinite exploration' condition is met. For SARSA, both GLIE conditions are needed. For both Q-learning and SARSA, the policy converges when both GLIE conditions are met. We can, therefore, summarize the results in the following table.

	GL	IE	Q (Q-learning)	Q (SARSA)	π (Q-learning)	π (SARSA)
1.	NO	YES	YES	NO	NO	NO
2.	YES	NO	NO	NO	NO	NO
3.	YES	NO	NO	NO	NO	NO
4.	YES	YES	YES	YES	YES	YES
5.	YES	YES	YES	YES	YES	YES

Question 40. (5 points)

Consider the following MDP with $\gamma = 0.8$, $r(5) = 100$, $r(\cdot) = 0$.



The initial matrix of Q-values is

$$\widehat{Q}(x,y) = \begin{bmatrix} - & - & - & - & 0 & - \\ - & - & - & 0 & - & 0 \\ - & - & - & 0 & - & - \\ - & 0 & 0 & - & 0 & - \\ 0 & - & - & 0 & - & 0 \\ - & 0 & - & - & 0 & 0 \end{bmatrix}.$$

Consider path $1-5-1-3$ and constant learning rate $\alpha = 0.1$. Show changes in Q values after the agent-environment interaction for the Q-learning algorithm.

[adapted from Richard Sutton's 609 course, see `http://www.incompleteideas.net/book/the-book-2nd.html`]

Answer:

After the first interaction, only value $Q(1,5)$ changes:

$$\widehat{Q}_1(1,5) = \widehat{Q}_0(1,5) + \alpha \cdot \left(r(5) + \gamma \cdot \max_{y'} \widehat{Q}_0(5,y') - \widehat{Q}_0(1,5) \right) = 0 + 0.1(100 + 0.8 \cdot \max\{0,0,0\} - 0) = 10.$$

The remaining interactions are analogous.

$$\widehat{Q}_2(5,1) = \widehat{Q}_1(5,1) + \alpha \cdot \left(r(1) + \gamma \cdot \max_{y'} \widehat{Q}_1(1,y') - \widehat{Q}_1(5,1) \right) = 0 + 0.1(0 + 0.8 \cdot \max\{10\} - 0) = \frac{8}{10},$$
$$\widehat{Q}_3(1,3) = \widehat{Q}_2(1,3) + \alpha \cdot \left(r(3) + \gamma \cdot \max_{y'} \widehat{Q}_2(3,y') - \widehat{Q}_2(1,3) \right) = 0 + 0.1(0 + 0.8 \cdot \max\{0,0,0\} - 0) = 0.$$

Question 41. (10 points)

Consider an active reinforcement learning algorithm implemented by SARSA or Q-learning.

1. (2 points) Unlike the temporal difference learning, SARSA and Q-learning algorithms learn Q values instead of U. Why is U not enough?

   Answer:

   The agent needs to know which action leads to the best reward. U requires knowledge of transition probabilities P. Therefore, the agent learns Q to avoid learning the transition model.

2. (3 points) Explain why those algorithms need to balance exploration vs. exploitation. What those terms mean, and which of those is preferred early in the learning.

   Answer:

   Exploration means probing suboptimal actions to find their values. Exploitation is the opposite; the agent greedily tries to maximize its reward. Both cannot be achieved at the same time. Therefore, the agent needs to start exploring so that it does not miss a good action due to a series of unfavorable events. On the contrary, once the values are reliably established, the agent can start to exploit to maximize its reward.

3. (2 points) SARSA and Q-learning are guaranteed to converge to an optimal policy if both:

   - convergence criteria for learning rate α known from TD-learning are met, and
   - convergence criteria on the explore-exploit policy are met.

   What are those criteria placed on the explore-exploit policy?

   Answer:

   The criteria placed on the explore-exploit policy are GLIE. Agent's policy must eventually become greedy in the limit with $k \to \infty$. The second assumption assumes that if a state is visited ∞-many times, then each action is tried in this state ∞-many times.

4. (1 point) Provide an example of an explore-exploit policy that guarantees policy convergence for SARSA and Q-learning.

   Answer:

   The GLIE conditions are fulfilled, for example, by the ε-greedy policy with

$$\varepsilon(n_x) \in \mathcal{O}\left(\frac{1}{n_x}\right).$$

5. (2 points) Will one of the algorithms learn Q-values even if one of the conditions is not met? If yes, which and why, if not, explain.

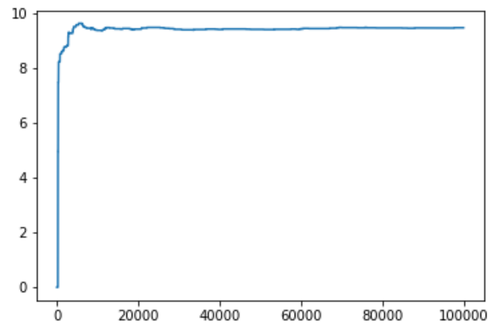
   Answer:

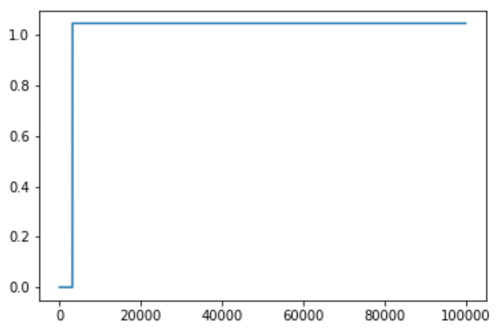
   SARSA, in general, as an on-policy algorithm, cannot learn the true values if one of the GLIE conditions is not met. However, the Q-learning can learn as an off-policy algorithm the Q-values even if the policy does not fulfill the 'greedy in the limit' condition. The algorithm won't act optimally, it will follow a suboptimal policy, but the Q-values will converge to the optimal as long as the 'infinite exploration' condition and learning rate convergence conditions are met.

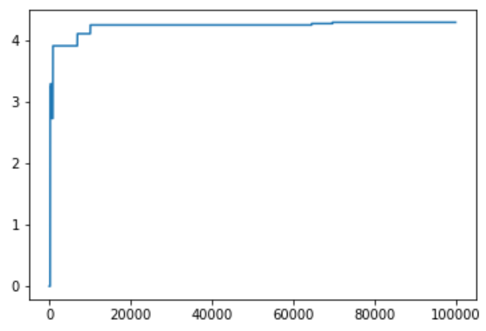
Question 42. (10 points)

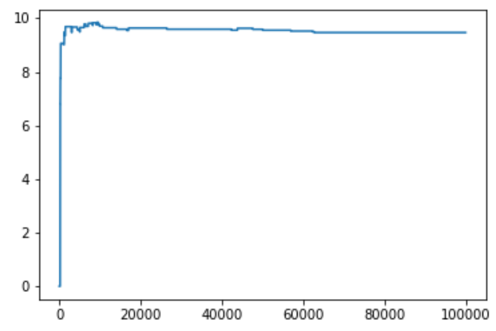
Consider an active reinforcement learning algorithm. You are not told whether it is an instance of SARSA or Q-learning. The implementation met all convergence criteria. All plots shown below are related to the same state-action pair Q-value, i.e., $\widehat{Q}(x, y)$. The action y is **suboptimal** in state x. The used explore-exploit policy was the ε-greedy policy, i.e., with probability ε a random action is selected; otherwise, the agent behaves greedily.

Now, consider four different situations of learning Q-values over 100 000 episodes.

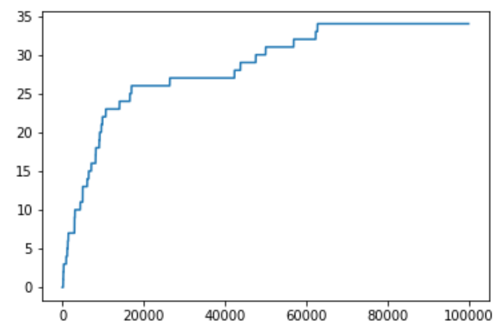

1.

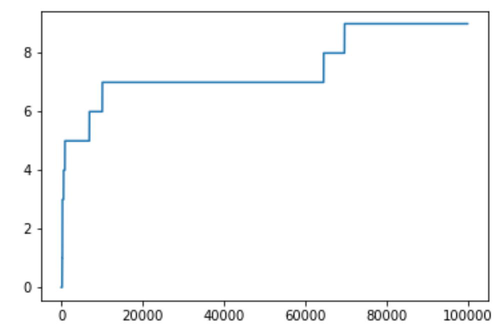

2.

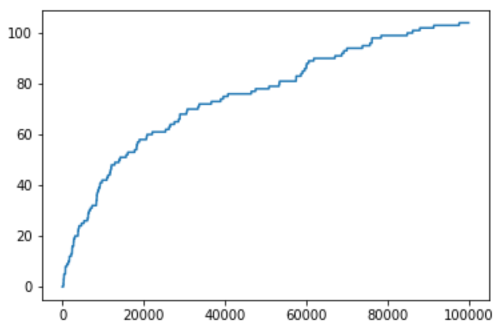

3.

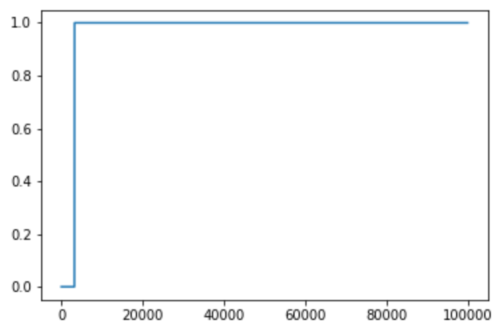

4.

1. (4 points) Four plots below show how many times action y was selected by the agent in state x. For example, point $(1000, 6)$ means that the action y was selected 6 times in state x over the first 1000 episodes.


a.


b.


c.


d.

Match figures a-d to figures 1-4. Explain your decision.

Answer:

For both SARSA and Q-learning, value $\widehat{Q}(x, y)$ changes only when state action pair x, y is visited. Therefore, d matches to 2, b matches to 3, a matches to 4, and c matches to 1.

2. (2 points) The ε was set as a function of the number of visits of state x. Relate the following four functions to the figures 1-4.

$$\text{i.} \quad \varepsilon(n_x) = \frac{8}{7+n_x} \qquad \text{ii.} \quad \varepsilon(n_x) = \frac{3}{2+n_x} \qquad \text{iii.} \quad \varepsilon(n_x) = \frac{100}{99+n_x} \qquad \text{iv.} \quad \varepsilon(n_x) = \frac{1000}{999+n_x}$$

Match those policies to figures 1-4. Explain your choice.

Answer:

The agent explores the least in case 2, which corresponds to function ii. We know that action y is suboptimal; therefore, under true Q-values it is chosen only when the agent decides to use random action. Therefore, by sorting by the ε value, we can conclude that case 3 corresponds to i, case 4 to iii, and case 1 to iv.

3. (1 point) Why should agents use different epsilon for different states. **Answer:**

The reasoning is the same as in the case of the learning rate. The number of visits is different for different states. Slow learning in states that are visited rarely should not influence the states that are visited often.

4. (2 points) Decide whether the learning algorithm used was SARSA or Q-learning. Explain your decision. **Answer:**

From the plots, it is more likely that the algorithm used was Q-learning. Figures 1 and 4 show very little difference even though in figure 4, the agent decides to use random action more often. This indicates that the algorithm is off-policy. SARSA would be influenced more by random decisions. However, as we do not know the environment, we cannot state this for sure. *In fact, Q-learning was used.*

5. (1 point) What is $Q(x, y)$? Explain your answer. **Answer:**

$Q(x, y)$ is very likely between 9 and 10, as figures 1 and 4 indicate. Figures 2 and 3 show insufficient learning. Figures 1 and 4 agree on this number.