Computability and Computational Complexity, A.Y. 2018–2019

Written exam

Mauro Brunato

Monday, June 17, 2019

Exercise 1

Write down a 1-tape Turing Machine with alphabet $\Sigma = \{\textvisiblespace, \mathtt{0}, \mathtt{1}\}$ which, upon execution on input $x \in \{\mathtt{0}, \mathtt{1}\}^*$ halts if and only if the number of ones and the number of zeroes in x have the same parity (i.e., they are both even or both odd).
Assume that the input string is delimited by infinite blanks on both sides, and that the computation begins at the leftmost symbol of x.

Exercise 2

Let $\mathcal{M}$ represent a Turing Machine, let $\lfloor\mathcal{M}\rfloor$ be any reasonable encoding of $\mathcal{M}$, and remember that, in our notation, $\mathcal{M}(x) = \infty$ means "$\mathcal{M}$ does not halt when executed on input x". Consider the following languages:

$$\begin{aligned}
L_1 &= \{\lfloor\mathcal{M}\rfloor \mid \exists x \mathcal{M}(x) \neq \infty\} = \{\lfloor\mathcal{M}\rfloor \mid \mathcal{M} \text{ halts on some inputs}\}\\
L_2 &= \{\lfloor\mathcal{M}\rfloor \mid \forall x \mathcal{M}(x) \neq \infty\} = \{\lfloor\mathcal{M}\rfloor \mid \mathcal{M} \text{ halts on all inputs}\}\\
L_3 &= \{\lfloor\mathcal{M}\rfloor \mid \exists x \mathcal{M}(x) = \infty\} = \{\lfloor\mathcal{M}\rfloor \mid \mathcal{M} \text{ doesn't halt on some inputs}\}\\
L_4 &= \{\lfloor\mathcal{M}\rfloor \mid \forall x \mathcal{M}(x) = \infty\} = \{\lfloor\mathcal{M}\rfloor \mid \mathcal{M} \text{ doesn't halt on any input}\}
\end{aligned}$$

2.1) Provide examples of TMs $\mathcal{M}_1, \dots, \mathcal{M}_4$ such that $\lfloor\mathcal{M}_1\rfloor \in L_1, \dots, \lfloor\mathcal{M}_4\rfloor \in L_4$.

2.2) Describe the set relationships between the four languages (i.e., which languages are subsets of others, which are disjoint, which have a non-empty intersection).

Hint — *For point 2.1, simple one- or two-state machines should suffice.*

Exercise 3

Let L be a language, and let $\mathcal{N}$ be a non-deterministic Turing Machine that decides $x \in L$ in time $O(|x|^3 \log |x|)$.

3.1) Suppose that, whenever $x \in L$, at least 15 computations of $\mathcal{N}(x)$ accept; what probabilistic complexity classes does L belong to, and why?

3.2) Suppose that, whenever $x \in L$, $\underline{\text{at most}}$ 15 computations of $\mathcal{N}(x)$ $\underline{\text{do not}}$ accept; what probabilistic complexity classes would L belong to, and why?

Hint — *Consider the following classes: **RP**, **coRP**, **ZPP**, **BPP**, **PP**. Bonus points if you also consider **P** and **NP**.*

Solution traces

Exercise 1

Write down a 1-tape Turing Machine with alphabet $\Sigma = \{␣, \texttt{0}, \texttt{1}\}$ which, upon execution on input $x \in \{\texttt{0}, \texttt{1}\}^*$ halts if and only if the number of ones and the number of zeroes in x have the same parity (i.e., they are both even or both odd).
Assume that the input string is delimited by infinite blanks on both sides, and that the computation begins at the leftmost symbol of x.

Solution trace

The most straightforward way to remember the parity of the number of zeroes and ones is to introduce 4 states:

- s_{ee} is the state corresponding to an even number of zeroes and an even number of ones;
- s_{eo} is the state corresponding to an even number of zeroes and an odd number of ones;
- s_{oe} is the state corresponding to an odd number of zeroes and an even number of ones;
- s_{oo} is the state corresponding to an odd number of zeroes and an odd number of ones.

The machine starts in state s_{ee} (no zeroes and no ones) and sweeps the input string until it finds a blank, indicating the end of the input. If, once a blank has been observed, the two parities are equal (i.e., the machine is in state s_{ee} or s_{oo}), then it halts; otherwise it enters a new state in which it runs forever, as per the specification:

	`0`	`1`	␣
s_{ee}	`0`, right, s_{oe}	`1`, right, s_{eo}	␣, right, `HALT`
s_{eo}	`0`, right, s_{oo}	`1`, right, s_{ee}	␣, right, s_{forever}
s_{oe}	`0`, right, s_{ee}	`1`, right, s_{oo}	␣, right, `HALT`
s_{oo}	`0`, right, s_{eo}	`1`, right, s_{oe}	␣, right, s_{forever}
s_{forever}	`0`, right, s_{forever}	`1`, right, s_{forever}	␣, right, s_{forever}

Observations

- There is a much simpler solution, obtained by collapsing the two states s_{ee} and s_{oo} in a single state, and by collapsing s_{oe} and s_{eo} in another state. This is equivalent to the observation that the two parities are equal if and only if the input x has even length.
- Of course, any representation of the TM was fine: as a transition table, an automaton graph…
- Unfortunately, many answers were undecipherable because of a lack of an explanation.

Exercise 2

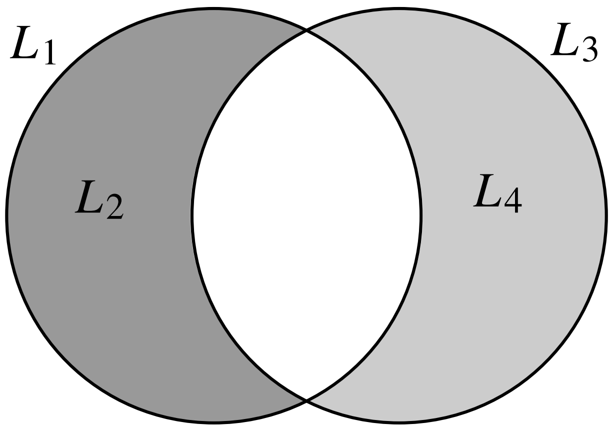
Let $\mathcal{M}$ represent a Turing Machine, let $\lfloor\mathcal{M}\rfloor$ be any reasonable encoding of $\mathcal{M}$, and remember that, in our notation, $\mathcal{M}(x) = \infty$ means "$\mathcal{M}$ does not halt when executed on input x". Consider the following languages:

$$\begin{aligned}
L_1 &= \{\lfloor\mathcal{M}\rfloor \mid \exists x \mathcal{M}(x) \neq \infty\} = \{\lfloor\mathcal{M}\rfloor \mid \mathcal{M} \text{ halts on some inputs}\} \\
L_2 &= \{\lfloor\mathcal{M}\rfloor \mid \forall x \mathcal{M}(x) \neq \infty\} = \{\lfloor\mathcal{M}\rfloor \mid \mathcal{M} \text{ halts on all inputs}\} \\
L_3 &= \{\lfloor\mathcal{M}\rfloor \mid \exists x \mathcal{M}(x) = \infty\} = \{\lfloor\mathcal{M}\rfloor \mid \mathcal{M} \text{ doesn't halt on some inputs}\} \\
L_4 &= \{\lfloor\mathcal{M}\rfloor \mid \forall x \mathcal{M}(x) = \infty\} = \{\lfloor\mathcal{M}\rfloor \mid \mathcal{M} \text{ doesn't halt on any input}\}
\end{aligned}$$

2.1 — Provide examples of TMs $\mathcal{M}_1, \dots, \mathcal{M}_4$ such that $\lfloor\mathcal{M}_1\rfloor \in L_1, \dots, \lfloor\mathcal{M}_4\rfloor \in L_4$.
2.2 — Describe the set relationships between the four languages (i.e., which languages are subsets of others, which are disjoint, which have a non-empty intersection).

Answer trace

2.1 — The machine that immediately halts ($s_0 = \texttt{HALT}$) is an example for L_1 and L_2. The machine that never halts (e.g., always moving right and staying in state s_0) is an example for L_3 and L_4.
2.2 — If a machine always halts, it clearly halts on some inputs; therefore, $L_2 \subset L_1$ (equality is ruled out by the fact that there are machines that halt on some inputs and don't on others: $L_1 \cap L_3 \neq \emptyset$). With similar considerations, we can say that $L_4 \subset L_3$.
L_2 is disjoint from both L_3.
Also, observe that $L_2 = L_1 \setminus L_3$ and $L_4 = L_3 \setminus L_1$.
The relationship among the sets can be shown in the following diagram:



Observations

This exercise has little to do with computability; not understanding what is being asked means a severe lack of logical bases. Sorry for the bluntness.

Exercise 3

Let L be a language, and let $\mathcal{N}$ be a non-deterministic Turing Machine that decides $x \in L$ in time $O(|x|^3 \log |x|)$.
3.1 — Suppose that, whenever $x \in L$, at least 15 computations of $\mathcal{N}(x)$ accept; what probabilistic complexity classes does L belong to, and why?
3.2 — Suppose that, whenever $x \in L$, <u>at most</u> 15 computations of $\mathcal{N}(x)$ <u>do not</u> accept; what probabilistic complexity classes would L belong to, and why?

Answer trace

3.1 — Clearly, $L \in$ **NP**, because a non-deterministic TM decides it in polynomial time, and therefore $L \in$ **PP** (but $\mathcal{N}$ must be tweaked in order to meet the definition). Observe that, if $x \in L$, the guaranteed ratio of accepting computations (which is a constant 15) to the total number tends to zero as the input size grows: the number of possible computations grows exponentially with the computation time. Therefore, there is no $\varepsilon > 0$ such that

$$\frac{15}{\text{Number of computations}} > \varepsilon;$$

this means that the existence of $\mathcal{N}$ alone does not guarantee that L belongs to any other probabilistic class (they all require a finite, nonzero bound).
3.2 — Again, $L \in$ **NP** for the same reason as above. This time, if $x \in L$, *almost all computations accept*: only a small, residual number (15 against an exponentially growing number) keep rejecting valid inputs. Since a very large fraction of computations (almost 100%) accepts valid inputs, and all invalid ones are rejected, the machine satisfies the definition of **RP**.

Observations

- Actually, in the case **3.2**, $L \in$ **P**; in fact, we just need to emulate $16 = 15+1$ computations of the NDTM (each being in polynomial time): if $x \in L$, in the worst case at least one of the computations will accept, otherwise all of them will reject. As a consequence, L belongs to *all* probabilistic classes that we defined.

- Saying "suppose that the total number of computations is 30, then the ratio is $1/2$" doesn't make sense: as said above, the number of computations is unbounded, and grows very quickly.

- $O(n^3 \log n)$ *is* polynomial, since $\log n = O(n)$.