Computability and Computational Complexity, A.Y. 2019–2020

Guide to the answers

Friday, July 3, 2020

Exercise 1

1.1) Let L be a language on a finite alphabet, and let $\mathcal{N}$ be a non-deterministic Turing machine on the same alphabet with the following properties:

- $\mathcal{N}(x)$ takes at most $|x|^2$ non-deterministic steps before halting ($|x|$ is the size of x).
- If $x \notin L$, then $\mathcal{N}(x)$ rejects the input.
- If $x \in L$, then $\mathcal{N}(x)$ accepts the input.

Are these properties sufficient for us to say that $L \in \mathbf{NP}$?

1.2) Suppose that $\mathcal{N}$ has the following additional property:

- At every step, $\mathcal{N}$ performs at most one binary non-deterministic choice (i.e., $\mathcal{N}$ has two transition functions)

Given this property and those listed in the previous point, determine an upper bound for the number $C_{\mathcal{N}}(x)$ of non-deterministic computations performed by $\mathcal{N}$ on input x as a function of the input size $|x|$.

1.3) Suppose that $\mathcal{N}$ has the following additional property:

- If $x \in L$, out of the $C_{\mathcal{N}}(x)$ computations of $\mathcal{N}(x)$, at least $\sqrt{C_{\mathcal{N}}(x)}$ end in an accepting state.

Is this additional property (and the previous ones) sufficient for us to say that $L \in \mathbf{RP}$?

Solution 1

1.1) Yes, the three properties are precisely the ones that define the class **NP** (Non-deterministic Polynomial). The first property ensures that $\mathcal{N}$ always halts within a polynomial number of steps with respect to the input size; the second and third property just say that $\mathcal{N}$ decides L.

1.2) A computation of $\mathcal{N}(x)$ takes at most $|x|^2$ steps. At every step, a computation can take two alternative paths, "branching" into two computations; i.e., the computational paths split into two (at most) at every step until completion, therefore ending in $2^{|x|^2}$ leaves.

1.3) Let $x \in L$ and $|x| = n$. All that we know about the ratio of accapting computations is

$$\frac{\text{number of accepting computations}}{\text{number of computations}} \geq \frac{\sqrt{C(n)}}{C(n)} = \frac{1}{\sqrt{C(n)}},$$

but this bound tends to zero as $C(n)$ increases. Therefore, there is no constant $\varepsilon > 0$ bounding the ratio from below. Another way to say it is that the ratio of accepting computations is vanishingly small as the input size increases. Therefore, the property isn't enough for us to say that $L \in \mathbf{RP}$.

Exercise 2

For each of the following properties of Turing machines, prove whether it is recursive or not. Whenever possible, use Rice's theorem.

2.1) M changes state at least once when executed on the empty input

2.2) M never remains on the same state for two consecutive steps

2.3) M accepts all inputs

Solution 2

2.1) The property is not semantic, therefore we cannot apply Rice's theorem. The property is clearly computable, as it only requires to understand if M is going to stay in the initial state indefinitely, or if it is going to move to a different state after a while, which we can obtain by analyzing the transition table or by simulating until we detect a repetition or a state change.

2.2) Again, the property is not semantic. However, every machine $\mathcal{M}$ can be transformed into a machine $\mathcal{M}'$ that changes state at every step by following this scheme:

- "split" every state s into an even state s_{even} and an odd state s_{odd}.
- Every transition rule moves from even to odd states and vice versa.

This way, we obtain a machine $\mathcal{M}'$ that performs the same computation as $\mathcal{M}$, but whenever $\mathcal{M}$ lingers on the same state s, $\mathcal{M}'$ alternates between states s_{even} and s_{odd}. Next, we can do the following:

- replace the halting state witn a dummy state that repeats indefinitely.

At this point, we have a machine $\mathcal{M}'$ that "remains on the same state for two consecutive steps" if and only if $\mathcal{M}$ halts. In other words, the stated property is equivalent to the halting problen, and is therefore uncomputable.

2.3) The property is semantic (it is shared by all and only those machines that decide the complete language), and it is not trivial (some machines, but not all, accept all strings). By Rice's theorem, the property is uncomputable.