Warm up exercise

Complexity Theory (Monsoon 2024)

Due on 5th August 2024 by 10 am

Note

- Submit your solutions to my office (M317), you can slide them under the door if it is locked. There are **no late-days** for this exercise.
- If you discuss the solutions with your peers, do credit them, and **write the final answers in your own words**. This also applies to using search engines, online forums or any other automated tools.
- You are not *required* to submit the solutions for this quiz to me, but it is strongly advised that you do so (and by the due date and time). It will help me in planning the initial lectures of the course accordingly.

Questions

1. **(Fast exponentiation)** Consider the following algorithm for exponentiation.

Input: Non-negative integers a, b, with $a \neq 0$.
Result: a^b

```
1 p ← a
2 while b > 1 do
3     p ← p · a      //multiply a into p
4     b ← b − 1
5 end
6 output p
```

   (a) Will the above algorithm output the correct answer for all valid inputs? If yes, then why; and if no, then can you explain what is wrong and fix it? **(5 points)**

   (b) How many multiplications does this (or the modified) algorithm make? Can you write an algorithm for exponentiation that makes about $\log b$ multiplications? **(10 points)**
   [Hint: You may assume that b is given as a string of bits.]

2. **(Existence of undecidable problems)** We will reconstruct an easy proof that not all decision problems can be solved by (single-tape) Turing machines.

   (a) Recall that a Turing machine is defined as follows, to be a "finite state automaton with access to unbounded memory".

   > For input alphabet $\Sigma = \{0, 1\}$ and tape alphabet $\Gamma = \{0, 1, \texttt{blank}\}$, a *Turing machine* is a 2-tuple (Q, δ) where Q is a finite set of states and δ is the *transition function* between the states given by
   >
   > $$\delta : Q \times \Gamma \to Q \times \Gamma \times \{\texttt{left}, \texttt{right}\}.$$
   >
   > We assume that the starting, accepting and rejecting states are fixed to be $q_0 = 0$, $q_{\texttt{accept}} = n - 2$ and $q_{\texttt{reject}} = n - 1$ respectively, without loss of any generality.

   Given the above definition, for a number n, roughly how many bits are sufficient to describe a Turing machine with n states ($|Q| = n$)? **(2 points)**

   (b) Using (a), what can be said about the "size" (more formally: cardinality) of the set of all possible Turing machines? **(3 points)**

   (c) Let us define a *decision problem* to be any list of expected `YES/NO` answers for every possible finite bit-string given as an input.

   For instance, if we say that any bit-string that is the binary representation of a prime number should produce a `YES` and all other strings should produce `NO`s, then we obtain the decision problem of testing primality.

   Along the lines of (b), what is the cardinality of the set of all possible decision problems? That is, when all finite strings are valid inputs? **(4 points)**

   (d) Using the above, show that there exist decision problems that cannot be solved by any Turing machine. **(1 points)**