Quiz 10A: Compute with Turing Machine

CS 212 Nature of Computation

Habib University — Fall 2023

Total Marks: 10 | Date: October 26, 2023
Duration: 10 minutes | Time: 830–840h

Student ID: _____

Student Name: _____

1. (10 points) So far, we have encountered Turing machines which solve decision problems, i.e., we are interested in whether they accept or reject an input. For other types of problems, we are interested in the content of the tape when the Turing machine halts. Let us call these, "compute" problems.

   We know that if a machine can perform the Boolean operations, then it can perform all the computations that we are used to with contemporary computers.

   Show that a Turing machine can compute the OR of two n-bit strings.

   That is, if the tape contains two n-bit strings, $a = a_1 a_2 \dots a_n$ and $b = b_1 b_2 \dots b_n$, separated by a blank symbol, a Turing machine can read the tape and halt leaving only an n-bit string, $c = c_1 c_2 \dots c_n$, on the tape. Each bit of c is the bitwise OR of the corresponding bits of a and b, i.e., $c_i = a_i \vee b_i$ for $1 \leq i \leq n$.

   Solution: We provide a proof by construction, i.e., we define a Turing machine for the computation.

   Proof. The machine zig-zags over the input to find corresponding bits in a and b and writes the output one call after the input. The input bits are blanked out (replaced with ␣) as they are processed. When the input is exhausted, the tape contains only the result surrounded by ␣.

   We provide an intermediate- or implementation- level description of the Turing machine. The machine starts with the head at a_1. We use "skip" to denote one or more moves of the head that do not replace any tape symbols.

   1. Remember the current symbol, call it a_i, and replace it with ␣.
   2. Move right (next bit of a to process).
   3. If the current symbol is ␣ (input is exhausted)
      (a) Skip right to find the first symbol that is 0 or 1 (last bit of b).
      (b) Remember the current symbol, call it b_i, and replace it with ␣.
      (c) Move right (end of input). Skip right to find ␣ (next free space for output).
      (d) Write $a_i \wedge b_i$.
      (e) Accept.
   4. If the current symbol is 0 or 1 (input remains to be processed)

(a) Skip right to find ␣ (end of a). Skip right to find the first symbol that is 0 or 1 (matching bit of b).

(b) Remember the current symbol, call it b_i, and replace it with ␣.

(c) Skip right to find ␣ (end of input). Skip right to find ␣ (next free space for output).

(d) Write $a_i \vee b_i$.

(e) Skip left to find ␣ (end of input). Skip left to find ␣ (just-processed bit of b). Skip left to find ␣ (separates a and b). Skip left to find ␣ (just-processed bit of a).

(f) Move right.

5. Repeat.

□