Practice Midterm 2 Solutions

1. Consider the language

\[ \text{INT}_{TM} = \{ \langle M_1, M_2 \rangle : L(M_1) \cap L(M_2) \neq \emptyset \}. \]

(Thus, \( \text{INT}_{TM} \) is the language associated with the problem of deciding whether, for two given Turing machines \( M_1 \) and \( M_2 \), there is some string that is accepted by both machines.)

(a) Show that \( \text{INT}_{TM} \) is Turing recognizable.
(b) Show that \( \text{INT}_{TM} \) is not decidable.
(c) Is the language

\[ \text{EINT}_{TM} = \{ \langle M_1, M_2 \rangle : L(M_1) \cap L(M_2) = \emptyset \}. \]

Turing recognizable?

(a) We construct a Turing machine that recognizes \( \text{INT}_{TM} \) as in the construction of an enumerator for a Turing-recognizable language. On input \( \langle M_1, M_2 \rangle \), for \( i = 1, 2, \ldots \), for each string \( s \) of length at most \( i \), simulate each \( M_1 \) and \( M_2 \) on input \( s \) for \( i \) steps, and accept if both \( M_1 \) and \( M_2 \) accept \( s \).

(b) We shall present a mapping reduction from \( A_{TM} \) to \( \text{INT}_{TM} \), and since \( A_{TM} \) is undecidable, it would follow that \( \text{INT}_{TM} \) is undecidable. The reduction is as follows: on input \( \langle M, w \rangle \), first construct a machine \( M_w \) that on input \( x \), check if \( x = w \). If so, it simulates \( M \) on \( x \) and otherwise, reject. In addition, construct a machine \( M_{all} \) that accepts all inputs. Output \( \langle M_w, M_{all} \rangle \). It is easy to see that \( M \) accepts \( w \) iff \( \langle M_w, M_{all} \rangle \in \text{INT}_{TM} \).

(c) Observe that \( \text{EINT}_{TM} \) is the complement of \( \text{INT}_{TM} \). Therefore, if \( \text{EINT}_{TM} \) is Turing recognizable, then together with the result in (a), we know that \( \text{INT}_{TM} \) is decidable, a contradiction. Hence, \( \text{EINT}_{TM} \) is not Turing recognizable.

2. Consider the following time-bounded variant of Kolmogorov complexity, written \( K_L(x) \), and defined to be the shortest string \( \langle M, w, t \rangle \) where \( t \) is a positive integer written in binary, and \( M \) is a TM that on input \( w \) halts with \( x \) on its tape within \( t \) steps.

(a) Show that \( K_L(x) \) is computable (by describing an algorithm that on input \( x \) outputs \( K_L(x) \)).

(b) Prove that for all positive integers \( n \), there exists a string \( x \) of length \( n \) such that \( K(x) = O(\log n) \) and \( K_L(x) \geq n \). (In fact, there is an algorithm that on input \( n \) finds such a \( x \).)

(a) Here’s an algorithm for computing \( K_L(x) \). On input \( x \),

1. Go through all binary strings \( s \) in lexicographic order, and for each such \( s \), parse \( s \) as \( \langle M, w, t \rangle \) for some TM \( M \), input \( w \) and integer \( t \). If \( s \) fails to parse, move to the next such \( s \).
2. Simulate \( M \) on input \( w \) for up to \( t \) steps. If it halts within \( t \) steps with \( x \) on its tape, output \( |s| \).
(b) By a counting argument, it is easy to see that for every \( n \), there exists a string \( x_n \) of length at least \( n \) such that \( K_L(x_n) \geq n \). Choose \( x_n \) to be the lexicographically first such string. Now, consider the machine \( T \) that on input an integer \( n \) written as a binary string, enumerates over all binary strings \( s \) in lexicographic order, computes \( K_L(s) \), and outputs the first \( s \) such that \( K_L(s) \geq n \). Then, \( T(n) = x_n \), so \( \langle T, n \rangle \) is a description for \( x_n \) and thus \( K(x_n) = O(\log n) \).

3. (Sipser 7.36) For a cnf-formula \( \phi \) with \( m \) variables and \( c \) clauses (that is, \( \phi \) is the AND of \( c \) clauses, each of which is an OR of several variables), show that you can construct in polynomial time an NFA with \( O(cm) \) states that accepts all nonsatisfying assignments, represented as Boolean strings of length \( m \). Conclude that the problem of minimizing NFAs (that is, on input a NFA, find the NFA with the smallest number of states that recognizes the same language) cannot be done in polynomial time unless \( P = NP \).

On input \( \phi \), construct a NFA \( N \) that nondeterministically picks one of the \( c \) clauses (via \( \epsilon \)-transitions), reads the input of length \( m \), and accepts if it does not satisfy the clause, and rejects otherwise. In addition, \( N \) also accepts all inputs of length not equal to \( m \). For each clause, we need \( O(m) \) states, so \( N \) has \( O(cm) \) states. It is clear that \( N \) can be computed in polynomial time. In addition, for any nonsatisfying assignment \( a \), at least one clause is not satisfied, so \( N \) accepts \( a \). Conversely, if \( N \) accepts \( a \), some clause is not satisfied, so \( a \) is a nonsatisfying assignment. Hence, \( N \) accepts all the nonsatisfying assignments of \( \phi \).

Next, suppose the problem of minimizing NFAs can be done in polynomial time. Then, consider the polynomial-time algorithm that on input a 3cnf formula \( \phi \) with \( m \) clauses, constructs a NFA \( N \) that accepts all the nonsatisfying assignments of \( \phi \). Observe that \( N \) accepts all binary strings iff \( \phi \) is not satisfiable. Now, run the NFA minimizing algorithm to produce a new NFA \( N' \). If \( N' \) contains exactly one state and accepts all binary strings, reject \( \phi \); otherwise, accept \( \phi \). This yields a polynomial-time algorithm for 3SAT, and hence \( P = NP \).

4. Prove that the halting problem \( HALT_{TM} \) for Turing machines is \( NP \)-hard.

It suffices to give a polynomial-time reduction from 3SAT to \( HALT_{TM} \). On input a 3cnf formula \( \phi \), output \( \langle M, \phi \rangle \), where \( M \) is a Turing machine that on input a 3cnf formula \( \psi \), enumerates over all possible assignments for \( \psi \), and halts if \( \psi \) is satisfied by at least one of the assignments, and loops otherwise.