CS21004 - Tutorial 12

Solution Sketch

1. Are the following problem decidable/undecidable?

   (a) $\{\langle B\rangle | B \text{ is a DFA that accepts a palindrome}\}$
   Solution: Yes, this is decidable. Given the DFA B, we can construct a DFA C that accepts the reverse language $\{w^R | w \in L(B)\}$. Now, we can construct a DFA for the intersection of the two languages, $L(B)$ and $L(C)$. We can now use emptyness testing. If this language is empty, B does not accept any palindrome, otherwise it does.

   (b) $LM_{TM} = \{\langle M, x\rangle | M \text{ ever moves left while computing on the input } x\}$
   Solution: Yes, this is decidable. Let k be the number of state of M and $|x|$ be the size of the input. We will know within $k + |x| + 1$ steps whether M makes a left move. Idea is that within first $|x|$ steps, if it does not make a left move, it is guaranteed to reach the first blank. Then, in the next k steps, if it does not take a left move, the input symbol is always the same (blank), and by pigeonhole principle, some input state will repeat, and hence the transition. In that case, we can guarantee that it is in a loop and will never make a left move.

2. Identify which of the following languages are decidable / undecidable.

   (a) $L_1 = \{M | M \text{ is a Turing machine that halts on exactly 481 strings}\}$
   Solution: This is not decidable. We can prove it by a reduction from MP.

   Let R be a TM that decides L_1. We can use R to create a TM S that decides MP as follows.

   S on input $< M, w >$, creates a new TM M_1 that does the following.

   M_1 on input x

   – If x is within the first 481 strings in lexicographical ordering, it accepts and halts.

   – Otherwise, if x is the 482nd string, runs M on w, and if M accepts w, M_1 accepts x and halts.

   – Otherwise, it goes into a trivial loop.

   Thus M_1 halts on 482 strings if M accepts w, otherwise M_1 halts on exactly 481 strings. Now, S runs R on M_1.

Thus

$< M, w >\in MP \to M_1$ halts on more than 481 strings $\to R$ accepts M_1

$< M, w >\notin MP \to M_1$ halts on exactly 481 strings $\to R$ rejects M_1

(b) $L_2 = \{M | M$ is a Turing machine and $|L(M)|$ is prime$\}$

Solution: Solution: This is not decidable. We can prove it by a reduction from MP.

Let R be a TM that decides L_2. We can use R to create a TM S that decides MP as follows.

S on input $< M, w >$, creates a new TM M_1 that does the following.

M_1 on input x

– If x is within the first 3 strings in lexicographical ordering, it accepts x.

– Otherwise, it runs M on w, and if M accepts w and x is the fourth string in the lexicographical ordering, it accepts, otherwise it rejects.

Thus $|L(M_1)|$ is 3 (prime) or 4 (non-prime) depending on whether M accepts w or not. Now, S runs R on M_1.

Thus

$< M, w >\in MP \to |L(M_1)|$ is prime $\to R$ accepts $\langle M_1, M_1\rangle$

$< M, w >\notin MP \to |L(M_1)|$ is not prime $\to R$ rejects $\langle M_1, M_1\rangle$

(c) $L_3 = \{\langle M_1, M_2\rangle\} | M_1$ and M_2 are two TMs, and $\epsilon \in L(M_1) \cup L(M_2)\}$

Solution: This is not decidable. We can prove it by a reduction from MP.

Let R be a TM that decides L_3. We can use R to create a TM S that decides MP as follows.

S on input $< M, w >$, creates a new TM M_1 that does the following.

M_1 on input x, runs M on w, and accepts if M accepts w. Thus $L(M_1)$ is ϕ or Σ^* depending on whether M accepts w or not. Now, S runs R on $\langle M_1, M_1\rangle$.

Thus

$< M, w >\in MP \to \epsilon \in L(M_1) \cup L(M_1) \to R$ accepts $\langle M_1, M_1\rangle$

$< M, w >\notin MP \to \epsilon \notin L(M_1) \cup L(M_1) \to R$ rejects $\langle M_1, M_1\rangle$

(d) $L_4 = \{\langle k, x, M_1, M_2, \ldots, M_k\rangle | k$ is a natural number, x is a string, M_i is a TM for all $1 \leq i \leq k$, and at least $k/2$ TM's of $M_1, M_2, \ldots, M_k$ halt on $x\}$

Solution: This is not decidable. We can prove it by a reduction from MP.

Let R be a TM that decides L_4. We can use R to create a TM S that decides MP as follows.

S on input $< M, w >$, creates a new TM M_1 that does the following.

M_1 on input w, runs M on w, accepts and halts if M accepts w. Now, S runs R on $\langle 2, w, M_1, M_1\rangle$.