

COMP330 Final Exam

Ryan Sowa, ID: 260886668

1. True or False? (Prove that your answer is correct). If L is a regular language and $A \subseteq L$, then A is decidable.

Answer:

False. Suppose by contradiction that the statement is true. Let the alphabet be denoted by Σ and let $L = \Sigma^*$ (regular as we can build a DFA for Σ^*). Since every undecidable language is a subset of Σ^* and we know that $A \subseteq L$, then A could be undecidable $\Rightarrow \Leftarrow$.

2. True or False? (Prove that your answer is correct). If L_1 is decidable and L_2 is Turing recognizable, then $L_1 \cap L_2$ is decidable.

Answer:

False. Suppose by contradiction that the statement is true. Let the alphabet be denoted by Σ , $L_1 = \Sigma^*$ (decidable as we can build a DFA for Σ^*), and $L_2 = A_{TM} = \{\langle M, w \rangle \mid M \text{ accepts } w\}$ (which is Turing Recognizable). Since $L_1 \cap L_2 = L_2 = A_{TM}$, which is undecidable, this is a contradiction $\Rightarrow \Leftarrow$. Thus, $L_1 \cap L_2$ need not be decidable.

3. True or False? (Prove that your answer is correct). If L_1 and L_2 are in NP, then $L_1 \cap L_2$ is in NP.

Answer:

True. Since L_1 and L_2 are in NP, then they can be decided by a non-deterministic Turing Machine in $O(n^c)$ and $O(n^k)$ time, respectively for some $c, k > 0$. To decide $L_1 \cap L_2$, we first check if the input $w \in L_1$ ($O(n^c)$ time). If it is not in L_1 , then reject. Else, check if $w \in L_2$ ($O(n^k)$ time). If it is not, then reject. Else, accept.

Since we can decide $L_1 \cap L_2$ in at most $O(n^c + n^k)$ time (polynomial) by a non-deterministic Turing Machine, $L_1 \cap L_2 \in NP$.

4. Describe the language of the following TM over the alphabet $\{0, 1, 2\}$: There are three states q_{start} , q_{accept} , and q_{reject} . Three arrows from q_{start} to itself with labels $\underline{1 \rightarrow 0, R}$ and $\underline{0 \rightarrow 1, L}$, and $\underline{2 \rightarrow 2, L}$. There is also one arrow from q_{start} to q_{accept} with label $\underline{\sqcup \rightarrow \sqcup, R}$.

Answer:

Because of the rule $\underline{2 \rightarrow 2, L}$, clearly, this language will not accept any strings containing 2 as the tape will never move to the right of this 2 to

find the blank symbol. Instead, the Turing Machine will loop forever on this input. Because the rule $\sqcup \rightarrow \sqcup, R$ leads to q_{accept} , it is also clear that the empty string will be accepted.

Next, notice that by the rule $0 \rightarrow 1, L$, a zero at the beginning of a string will be counted as a 1 (the tape head still points at this cell). Whenever we move to the right, by the rule $1 \rightarrow 0, R$, we will need to have to have seen a 1, which we change to a 0. Therefore, we can assume that at all times, there will be only 0's to the left of the tape head.

If we see a 1, we can change this 1 to a 0 and keep moving towards the end of the string.

If we see a 0 at some position m relative to the leftmost cell of the tape, we change this 0 to a 1 and move to the left. Since there are only $m - 1$ 0's to the left of the tape head, we must see a 0. We change all these 0's to 1's and move all the way to the beginning of the tape where we change the first cell to a 1 and cannot move left (so stay at the first cell). From here, we use the m 1's we wrote to get to position $m + 1$.

In both cases (of seeing a 1 and a 0), we are eventually able to move right of a given configuration, so given any string with 0's and 1's (and no 2's), eventually, we will end up past the end of the string where we will find the \sqcup symbol. Since the rule $\sqcup \rightarrow \sqcup, R$ leads to q_{accept} , we will accept.

Therefore, the language of this TM is $\{0, 1\}^*$.

5. Rigorously establish the decidability or undecidability of the following language:

$$L = \{\langle M_1, M_2 \rangle \mid M_1, M_2 \text{ are TM's and } L(M_1) \subseteq L(M_2)\}.$$

Answer:

Suppose that L is decidable. Then, there exists some Turing Machine R that decides it.

On input $\langle M \rangle$ to $E_{TM} = \{\langle M \rangle \mid M \text{ is a TM that does not accept any } w\}$, construct a Turing Machine N which does the following: On input x , reject x . The language of N is, therefore, \emptyset . Then, use the Universal Turing Machine U to run R on $\langle M, N \rangle$. If R accepts, this means that $L(M)$ was a subset of $L(N) = \emptyset$, so $L(M) = \emptyset$ must be true. Thus, accept M to E_{TM} . If R rejects, that means that $L(M)$ was not a subset of $L(N) = \emptyset$. Thus, $L(M) \neq \emptyset$, so reject M to E_{TM} .

Since we have shown that if L is decidable, E_{TM} is decidable, this is a contradiction (E_{TM} is undecidable) $\Rightarrow \Leftarrow$. Thus, L must be undecidable.

6. Let us call a deterministic Turing Machine M super-fast if there exists a constant c (here c can depend on M but it does not depend on the input) such that the following holds: On every input w , the TM M halts after at most c steps. Rigorously establish the decidability or undecidability of the following language:

$$L = \{\langle M \rangle \mid M \text{ is a super-fast TM}\}.$$

Answer:

If M halts in at most c steps on all possible strings of length c , then it will halt in at most c steps on all inputs (as any string longer than c will contain a substring we can halt on).

If the input is of the wrong form, then reject. On input $\langle M \rangle$ to L , check if M halts on every possible string over the alphabet Σ of length c in at most c steps. If M does not halt in at most c steps on one of these strings, then reject.

If M has not rejected after checking each of these strings, then accept.

Therefore, L is decidable.

7. Let L be the set of all $\langle p \rangle$ such that p is a multivariate polynomial with integer coefficients that evaluates to zero for some assignment of positive integers to its variables. For example, $\langle x_1^2 + x_2^2 - 5 \rangle \in L$ as it evaluates to 0 if we set $x_1 = 1$ and $x_2 = 2$. On the other hand, $\langle x_1^2 - 5 \rangle$ is not in L . Prove that L can be decided by an oracle Turing Machine that has access to an oracle for

$$Halt_{TM} = \{\langle M, w \rangle \mid M \text{ is a TM that halts on input } w\}.$$

Answer:

If the input is of the incorrect form, then reject. On input $\langle p \rangle$ to L , design a Turing Machine N which can be described as follows: On input $\langle p \rangle$, plug in all possible permutations of values for the indeterminates of p and calculate the result. If one of these permutations equals 0, then accept. Else, continue calculating permutations.

Use the oracle to check if $\langle N, p \rangle \in Halt_{TM}$. If $Halt_{TM}$ accepts, then this means we were able to find some permutation of p which resulted in 0, so accept. Else, we were unable to find a permutation which resulted in 0, so reject.

Therefore, L can be decided by an oracle for $Halt_{TM}$.

8. Consider the following language:

$$L = \{\langle M, w \rangle \mid M \text{ is a TM and } L(M) = \{w\}\}.$$

- (a) Is L Turing recognizable? (Prove your answer)

Answer:

No, L is not Turing Recognizable. Suppose by contradiction that L is Turing Recognizable.

Then, on input $\langle M, w \rangle$ to A_{TM}^C , construct a Turing Machine N .

N does the following: Let α be any symbol from the alphabet. On input x , if $x = \alpha$, then accept. If $x \neq \alpha$, then run M on w . If M accepts, then accept. If M rejects or loops, then reject or loop (respectively).

Since

$$\langle M, w \rangle \in A_{TM}^C \iff \langle N, \alpha \rangle \in L$$

, we can infer that a Turing Machine R that recognizes L can be used to recognize A_{TM}^C , which is NOT Turing Recognizable $\Rightarrow \Leftarrow$. Therefore, L must not be Turing Recognizable.

- (b) Is the complement of L Turing recognizable? (Prove your answer)

Answer:

No, $\bar{L} = \{\langle M, w \rangle \mid M \text{ is a TM and } L(M) \neq \{w\}\}$ is not Turing Recognizable. Suppose by contradiction that \bar{L} is Turing Recognizable.

Then, on input $\langle M, w \rangle$ to A_{TM}^C , construct a Turing Machine N .

N does the following: On input x to N , if $x = w$, then run M on w . If M accepts, then accept. If M rejects or loops, then reject or loop (respectively). If $x \neq w$, then reject.

Since

$$\langle M, w \rangle \in A_{TM}^C \iff \langle N, w \rangle \in \bar{L}$$

, we can infer that a Turing Machine R that recognizes \bar{L} can be used to recognize A_{TM}^C , which is NOT Turing Recognizable $\Rightarrow \Leftarrow$. Therefore, \bar{L} must not be Turing Recognizable.