Automata Theory - Final (Solutions)

K. Subramani
LCSEE,
West Virginia University,
Morgantown, WV
{ksmani@csee.wvu.edu}

1 Problems

1. **Diagonalization:** The Halting problem is defined as follows: Given a Turing Machine $M = \langle Q, \{0, 1\}, \Gamma, \delta, q_0, \Box, F\rangle$ and a string $w \in \Sigma^*$, determine whether M halts on w. In class, we proved that the Halting problem is undecidable using two different techniques. The first technique was constructive, where we constructed a series of Turing Machines, which led to a contradiction. The second technique was based on the observation that if the Halting problem is decidable then all recursively enumerable languages would become recursive. In this question, I am asking you to prove that the Halting problem is undecidable using diagonalization.
Hint: Recall that the set of Turing Machines is countable and construct the table of Turing Machines presented with Turing Machines.

   Solution: Assume that the Halting Problem is decidable, i.e., there exists an algorithmic procedure $\mathcal{A}$, which when presented with a Turing Machine and a string, unambiguously determines whether or not M halts on w.

   In class, we argued that the set of Turing Machines over a fixed alphabet is countable and therefore denumerable. Let $\mathcal{M} = \{M_1, M_2, \ldots\}$ denote some ordering of the set of all Turing Machines over the alphabet $\{0, 1\}$. We also showed that every Turing Machine is basically a binary string; we use w_i to denote the binary string representing Turing Machine M_i.

   We construct the table $\mathbf{R}$, where $R[i, j] = 0$, if the Turing Machine M_i halts on the binary string w_j and is 1 otherwise. Since, the Halting Problem is decided by $\mathcal{A}$, each entry in $\mathbf{R}$ is unique and can be computed.

   Based on table $\mathbf{R}$, we can construct the following Turing Machine M': M' halts on w_i, if and only if M_i does not halt on w_i. Clearly M' is a valid Turing Machine and hence it must appear at some point in the enumeration $\mathcal{M}$. Assume that $M' = M_k$ for some index k.

   What is $R[k, k]$? If $R[k, k] = 0$, then it means that Turing Machine M_k does halt on w_k; however, this would mean that M_k should not halt on w_k, by the definition of M_k. Likewise, if $R[k, k] = 1$, then it means that M_k does not halt on w_k; however, this forces M_k to halt on w_k!

   The above contradiction resulted because we assumed the existence of an algorithmic procedure that would unambiguously decide the Halting Problem. We can therefore conclude that the Halting Problem is undecidable. $\Box$

2. **Countability:**

   (a) The set $N = \{1, 2, 3, \ldots\}$ is known to be a countable set. Is the set $N \times N$ countable? (2 points.)

   (b) Let $I = (0, 1)$ and let $\Re^+ = (0, \infty)$. Which set has more elements? (2 points.)

   Solution:

   (a) The function $f : N \to Q^+$, defined by $f(a, b) = \frac{a}{b}$, puts the set $N \times N$ in a one-to-one correspondence with the set Q^+. Hence, the cardinalities of the two sets are equal and since we proved Q^+ to be countable in class, it follows that the set $N \times N$ is also countable.

(b) Clearly $|I| \leq |\Re^+|$, since every element of I is also an element of $\Re^+$. Now, consider the function $f : \Re^+ \rightarrow I$ defined by:

$$f(x) = \frac{1}{1+x}$$

It is not hard to see that f is injective, i.e., distinct elements in $\Re^+$ are mapped into distinct elements of I. It therefore follows that $|\Re^+| \leq |I|$, from which we can conclude that the two sets have the same cardinality.

This should not be surprising since the set of even numbers has the same cardinality has the set of natural numbers, i.e., in case of infinite sets, it is possible for a set A to be a proper subset of a set B and yet have the same cardinality as B.

$\Box$

3. **Language Theory:** In class we laboriously argued that the concept of undecidability did not apply to problems which are characterized by a single instance. Let us say that a problem P has three instances. Can such a problem be undecidable?

   Solution: Let us call the three instances I_1, I_2 and I_3. Note that there are precisely 8 possibilities for these instances, with respect to whether they are "yes" instances or "no" instances. For instance, one possibility is that I_1 is a "yes" instance, while the other two are not.

   Accordingly, we could construct eight distinct Turing Machines, each of which declares a distinct subset of $\{I_1, I_2, I_3\}$ to be "yes" instances. At least one of the eight Turing Machines produces the correct answer; thus although we may not know how to solve the problem P, there exists a Turing Machine which correctly decides it. It follows that problem P cannot be undecidable,as per the definition of undecidability. $\Box$

4. **Undecidability:** The Total-Halting problem is defined as follows: Given a Turing Machine $M = \langle Q, \Sigma, \Gamma, \delta, q_0, \Box, F \rangle$, determine whether M halt on all inputs $w \in \Sigma^*$. Is the Total-Halting Problem decidable?

   Solution: Assume that the Total-Halting problem is decidable and that there exists and algorithm $\mathcal{A}$, which when given a Turing Machine $M = \langle Q, \Sigma, \Gamma, \delta, q_0, \Box, F \rangle$ unambiguously and correctly decides whether M halts on all inputs $w \in \Sigma^*$.

   We now reduce the Halting Problem to the Total-Halting problem. Recall that in the Halting Problem, we are given a Turing Machine $M = \langle Q, \Sigma, \Gamma, \delta, q_0, \Box, F \rangle$ and a string $w \in \Sigma^*$ and asked whether M halts on w. Observe that given an instance $\langle M, w \rangle$ of the Halting Problem, we can construct the Turing Machine M', which behaves as follows: If the input to M' is exactly w, then M' simulates the execution of M on w; for any other string M' simply halts. It is not hard to see that M halts on w, if and only if, M' halts on all inputs! Since the Total-Halting problem is decidable, we could present the Turing Machine M' to the algorithm A and thereby solve the Halting Problem instance. Since all Halting Problem instances could be thus decided, it follows that the Halting Problem is decidable as well, which is absurd.

   We therefore conclude that the Total-Halting problem is undecidable. $\Box$

5. **Properties of Regular Languages:** Let L_1 and L_2 denote two languages over an alphabet Σ. We define $cor(L_1, L_2)$ as follows:

   $$cor(L_1, L_2) = \{w \in \Sigma^* \ : \ w \in L_1^c \text{ or } w \in L_2^c\}$$

   where L^c denotes the complement of language L. Show that $cor(L_1, L_2)$ is regular, if L_1 and L_2 are regular.

   Solution: Since L_1 is regular, so is L_1^c; likewise, the regularity of L_2^c follows from the regularity of L_2. In class, we showed that the union of two regular languages is also regular and thus $L_1^c \cup L_2^c$ is also regular. But this is precisely the language $cor(L_1, L_2)$ and hence, we conclude that $cor(L_1, L_2)$ is regular, if L_1 and L_2 are. $\Box$