CSL253 - Theory of Computation

TUTORIAL -7

Team Members

1. NDS GANESH - 12341500
2. V KARTHIK - 12342370
3. K CHARITH REDDY - 12341040

Question 15

Question 15

Let $S = \{M \mid M \text{ is a DFA and if } M \text{ accepts } w, \text{ then } M \text{ also accepts } w^R\}$. Show that S is decidable.

Solution Overview

Let M be a DFA. We are given that M accepts w^R whenever it accepts w. This implies that the language $L(M)$ is closed under reversal, i.e., $L(M) = L(M^R)$, where M^R is a DFA that accepts the reverse of the strings accepted by M.

We construct a Turing machine T that decides S as follows:

$T =$ "On input M, where M is a DFA:

1. Construct a DFA N that accepts $L(M)^R$ (the reversal of the language accepted by M).
2. Submit (M, N) to the decider for EQ_{DFA} (the problem of testing whether two DFAs are equivalent).
3. If the decider for EQ_{DFA} accepts, then accept.
4. If it rejects, then reject.

Since the construction of N from M and the procedure for checking DFA equivalence are both decidable, the Turing machine T is a decider. Therefore, the language S is decidable.

Construction 1 (Reversal DFA)**.** *To construct N that accepts $L(M)^R$, we proceed as follows:*

- *Convert the DFA M to an equivalent NFA.*

- *Reverse all the transitions in the NFA.*
- *Make the start state a new accepting state.*
- *Create a new start state with ϵ-transitions to the former accepting states.*
- *Convert the resulting NFA into an equivalent DFA N.*

This N accepts the reverse of every string in $L(M)$.

Proof

To prove this theorem, we use Theorem 4.4. We construct a new DFA C from A and B, where C accepts only those strings that are accepted by either A or B but not by both. Thus, if A and B recognize the same language, C will accept nothing. The language of C is

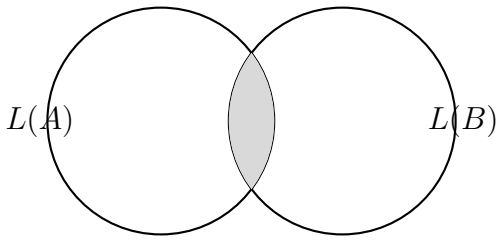
$$L(C) = \left(L(A) \cap \overline{L(B)}\right) \cup \left(\overline{L(A)} \cap L(B)\right).$$

This expression is sometimes called the *symmetric difference* of $L(A)$ and $L(B)$ and is illustrated in the following figure. Here, $\overline{L(A)}$ is the complement of $L(A)$. The symmetric difference is useful here because $L(C) = \emptyset$ iff $L(A) = L(B)$.

We can construct C from A and B using constructions for complementation, union, and intersection. These are algorithmic steps executable by a Turing machine. Once we have constructed C, we can use Theorem 4.4 to test whether $L(C)$ is empty. If it is, then $L(A) = L(B)$.

F = "On input $\langle A, B \rangle$, where A and B are DFAs:

1. Construct DFA C as described.
2. Run TM T from Theorem 4.4 on input $\langle C \rangle$.
3. If T accepts, *accept*. If T rejects, *reject*."



Conclusion 1. *Hence, by constructing N and comparing it with M using a decider for DFA equivalence, we can decide whether $L(M) = L(M^R)$. Therefore, the language S is decidable.*

Question 21

Question 21 (Balanced DFA)

Let

BALDFA = $\{(M) \mid M$ is a DFA that accepts some string contains an equal no.of 0s,1s$\}$.

Show that BALDFA is decidable. *(Hint: Theorems about CFLs are helpful here.)*

Given:

M is a DFA, and it accepts a string that contains an equal number of 0s and 1s.

Proof:

To prove that BALDFA is decidable, we use the concept of Context-Free Languages (CFLs).

Consider the language $E = \{w \mid w$ contains an equal number of 0s and 1s$\}$. This language E is a well-known context-free language. It can be generated by a context-free grammar.

We are interested in whether the language accepted by the DFA M, denoted by $L(M)$, has a non-empty intersection with E. That is, we want to determine if $L(M) \cap E \neq \emptyset$.

It is a fundamental result that the intersection of a regular language (like $L(M)$) and a context-free language (like E) is also a context-free language.

Therefore, $L(M) \cap E$ is a CFL.

We know that the emptiness problem for context-free languages is decidable. There exists a Turing machine that can take a CFG as input and determine whether the language generated by that grammar is empty or not.

Construction of a Decider for BALDFA:

We can construct a Turing machine T that decides BALDFA as follows:

1. On input $\langle M \rangle$, where M is a DFA:
2. Construct a Pushdown Automaton (PDA) P_E that recognizes the language $E = \{w \mid \#_0(w) = \#_1(w)\}$. (Such a PDA exists).
3. Construct a PDA $P_{intersection}$ that recognizes the intersection $L(M) \cap L(P_E) = L(M) \cap E$. The construction for the intersection of a regular language and a CFL to obtain a CFL (and thus a PDA) is a standard result in automata theory.
4. Use a decider for the emptiness problem of CFLs. Let this decider be $D_{CFL-EMPTY}$. Run $D_{CFL-EMPTY}$ on the PDA $P_{intersection}$.
5. If $D_{CFL-EMPTY}$ accepts (meaning $L(P_{intersection})$ is not empty), then accept $\langle M \rangle$. This indicates that M accepts at least one string with an equal number of 0s and 1s.

6. If $D_{CFL-EMPTY}$ rejects (meaning $L(P_{intersection})$ is empty), then reject $\langle M \rangle$. This indicates that M does not accept any string with an equal number of 0s and 1s.

Since each step of this construction is algorithmic and the emptiness of CFLs is decidable, the Turing machine T is a decider for BALDFA.

Conclusion:

Therefore, BALDFA is a decidable language.