CSCI 2210: Theory of Computation

Problem Set 6 (due 11/08)

Student: Rafael Almeida   *Collaborators:* Cal Thompson

Problem 1. Show that the following languages are decidable.

(a) $C_{regex} = \{\langle R, S\rangle \mid R \text{ and } S \text{ are regular expressions and } L(R) \subseteq L(S)\}$

M = "on input R, S:

1. Construct DFA D_R from R
2. Construct DFA D_S from *complement* of S
3. Intersect D_R and D_S as D_I
4. Transition through the states of D_I
    5. If one is accepting, **reject**
6. **Accept**"

If L(R) is a subset of L(S), L(R) intersected with the complement of L(S) should be empty. If not, that means there is a string in L(R) not in L(S).

(b) $\mathsf{EQ}_{\mathsf{DFA/REX}} = \{\langle D, R\rangle \mid D \text{ is a DFA, } R \text{ is regular expression, and } L(D) = L(R)\}$

M = "on input D, R:

1. Convert R into a DFA D_R
2. Minimize D and D_R
3. Transition through the states of D and D_R
    4. If one is not equivalent, **reject**
5. **Accept**"

Because minimal DFAs are unique, if the minimal version of D and D_R are not equivalent all throughout, it means that L(D) ≠ L(R)

Problem 2. Consider the following facts:

- Regular languages are closed under intersection.
- Regular expression $\Sigma^* w \Sigma^*$ describes all strings over alphabet Σ that have w as a substring.

Show that language L is decidable.

$$L = \{\langle D, u\rangle \mid D \text{ is a DFA and there exists } w \in L(D) \text{ such that } u \text{ is a substring of } w\}$$

M = "on input D, u:

1. Create DFA for regular expression Σ*uΣ*, D_SUB
2. Intersect D with D_SUB, D_INTER
3. Transition through the states of D_INTER
    4. If an accept state is found, **accept**
5. **Reject**"

D_INTER recognizes the language of D and of strings that contain u as a substring. Therefore, if D_INTER accepts some string, that string is accepted by D and has u as a substring.

Problem 3. Consider languages $L_1 \subseteq L_2 \subseteq L_3$. Knowing that L_1 and L_3 are decidable, can one conclude that L_2 is also decidable? Justify your answer.

No. Consider L1 being the empty set and L3 being Σ* which are both decidable. L2 could be A_TM, which is known to be undecidable.

Problem 4. Show that the following languages are undecidable.

(a) $\mathsf{REV}_{\mathsf{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = (L(M))^R\}$. *Recall that* $L^R = \{w^R \mid w \in L\}$.

M_P = "on input x:
1. If M accepts w, **accept** palindromes
2. Else, **reject** palindromes"

S = "on input M, w:
1. Use the description of M, w to construct M_P
2. Run TM_REV on input M_P
3. If accepts, **accept**.
4. Else, **reject**"

Above is a decider for A_TM out of a decider for REV_TM, TM_REV. If a language is equal to its reverse, its strings are palindromes. Therefore, M_P can be made to convert instances of A_TM to REV_TM by accepting or rejecting palindromes. If TM_REV exists, it can be used to decide A_TM as seen above. However, this is a contradiction since A_TM is known to be undecidable. Thus, the initial assumption that TM_REV exists is false, so REV_TM is undecidable.

(b) $\mathsf{HALT}_{\mathsf{any}} = \{\langle M \rangle \mid M \text{ is a TM that halts on at least one string}\}$.

M_H = "on input x:
1. If M accepts w, always **halt**
2. Else, always **loop**"

S = "on input M, w:
1. Use the description of M, w to construct M_H
2. Run TM_HALT on M_H
3. If accepts, **accept**. Else, **reject**"

Above is a decider for A_TM out of a decider for HALT_ANY, TM_HALT. M_H can be made to convert instances of A_TM to HALT_ANY by always halting or looping. If TM_HALT exists, it can be used to decide decide A_TM as seen above. This is a contradiction since A_TM is known to be undecidable. Thus, the initial assumption that TM_HALT exists is false, so HALT_ANY is undecidable.

Problem 5. Show that the semidecidable languages are closed under intersection but NOT closed under complement.

Intersection

$$
\begin{aligned}
M_1 &= (Q_1, \Sigma, \Gamma_1, \delta_1, q_1^0, q_1^a, q_1^r) \\
M_2 &= (Q_2, \Sigma, \Gamma_2, \delta_2, q_2^0, q_2^a, q_2^r) \\
M_\cap &= (Q_\cap = Q_1 \times Q_2, \\
&\Sigma, \\
&\Gamma_\cap = \Gamma_1 \times \Gamma_2, \\
&\delta_\cap = (Q_1 \times Q_2) \times \Gamma^- \to (Q_1 \times Q_2) \times \\
&\Gamma_\cap \times \{R, L\}, \; where\, \Gamma^- = \Gamma_\cap - \{q_\cap^a, q_\cap^r\}, \\
&q_\cap^0 = (q_1^0, q_2^0) \\
&q_\cap^a = (q_1^a, q_2^a) \\
&q_\cap^r = (q_1^r, q_2^r))
\end{aligned}
$$

Complement

Semidecidable languages are not closed under complement because it would lead to a contradiction. Consider semidecidable language L and its TM, TM_L. If semidecidable languages are closed under complement, then the complement of L, L_C has a TM, TM_L_C. As seen before, TM_L and TM_L_C can be used to create a TM_D that decides L by simulating TM_L and TM_L_C for an increasing number of steps until one of them halts. The existence of TM_D contradicts the fact that L is semidecidable. Therefore TM_D does not exist. Thus, our assumption that semidecidable languages are closed under complement is false.

Problem 6. Show that if L is semidecidable and $L \leq_m \overline{L}$, then L is decidable.

If L is semidecidable, then there is a $TM\ M_1$ where $L(M_1) = L$

If $L \leq_m \bar{L}$ then there is a function f where $w \in L \iff f(w) \in \bar{L}$ and $\bar{L}$ is semidecidale

If $\bar{L}$ is semidecidable, then there is a $TM\ M_2$ where $L(M_2) = \bar{L}$

Therefore, a decider for L, M_D can be defined as

M_D = "on input w:

1. i = 1
2. Run the following machines for i steps
   3. M_1 on input w
      4. If accepts, **accept**. If rejects, **reject**
   5. M_2 on input f(w)
      6. If accepts, **reject**. If rejects, **accept**
7. Increment i

Since M_2 accepts the complement of M_1, both can be used to create M_D, a halting TM for L. M_D is halting because if w is in L, step 4 makes M_D accept; if w is not in L and M_1 halts, step 4 makes M_D reject; if w is not in L and M_1 loops, we know f(w) M_2 will accept, so step 6 rejects it. Therefore, L is decidable.