CS/ECE 374 A ✧ Spring 2018

Homework 3

Due Tuesday, February 13, 2018 at 8pm

---

1. Describe context-free grammars for the following languages over the alphabet $\Sigma = \{\mathtt{0}, \mathtt{1}\}$. For each non-terminal in your grammars, describe in English the language generated by that non-terminal.

   (a) $\left\{\mathtt{0}^a\mathtt{1}\mathtt{0}^b\mathtt{1}\mathtt{0}^c \;\middle|\; a+b=c\right\}$

   (b) $\left\{w \in (\mathtt{0}+\mathtt{1})^* \;\middle|\; \#(\mathtt{0},w) \leq 2\cdot\#(\mathtt{1},w)\right\}$

   (c) Strings in which the substrings `00` and `11` appear the same number of times. For example, $\mathtt{1100011} \in L$ because both substrings appear twice, but $\mathtt{01000011} \notin L$. *[Hint: This is the complement of the language you considered in HW2.]*

2. Let $inc\colon \{\mathtt{0},\mathtt{1}\}^* \to \{\mathtt{0},\mathtt{1}\}^*$ denote the *increment* function, which transforms the binary representation of an arbitrary integer n into the binary representation of $n+1$, truncated to the same number of bits. For example:

   $$inc(\mathtt{0010}) = \mathtt{0011} \qquad inc(\mathtt{0111}) = \mathtt{1000} \qquad inc(\mathtt{1111}) = \mathtt{0000} \qquad inc(\varepsilon) = \varepsilon$$

   Let $L \subseteq \{\mathtt{0},\mathtt{1}\}^*$ be an arbitrary regular language. Prove that $inc(L) = \{inc(w) \mid w \in L\}$ is also regular.

3. A ***shuffle*** of two strings x and y is any string obtained by interleaving the symbols in x and y, but keeping them in the same order. For example, the following strings are shuffles of `HOGWARTS` and `BRAKEBILLS`:

   `HOGWARTSBRAKEBILLS` `HOGBRAKEWARTSBILLS` `BHROAGKWEABRITLSLS`

   More formally, a string z is a shuffle of strings x and y if and only if (at least) one of the following conditions holds:

   - $x = \varepsilon$ and $z = y$
   - $y = \varepsilon$ and $z = x$
   - $x = ax'$ and $z = az'$ where z' is a shuffle of x' and y
   - $y = ay'$ and $z = az'$ where z' is a shuffle of x and y'

   For any two languages L and L' over the alphabet $\{\mathtt{0},\mathtt{1}\}$, define

   $$shuffles(L, L') = \left\{z \in \{\mathtt{0},\mathtt{1}\}^* \;\middle|\; z \text{ is a shuffle of some } x \in L \text{ and } y \in L'\right\}$$

   Prove that if L and L' are regular languages, then $shuffles(L, L')$ is also a regular language.

Solved problem

4. (a) Fix an arbitrary regular language L. Prove that the language $\mathit{half}(L) := \{w \mid ww \in L\}$ is also regular.

Solution: Let $M = (\Sigma, Q, s, A, \delta)$ be an arbitrary DFA that accepts L. We define a new NFA $M' = (\Sigma, Q', s', A', \delta')$ with ε-transitions that accepts $\mathit{half}(L)$, as follows:

$$Q' = (Q \times Q \times Q) \cup \{s'\}$$
$$s' \text{ is an explicit state in } Q'$$
$$A' = \{(h, h, q) \mid h \in Q \text{ and } q \in A\}$$

$$\delta'(s', \varepsilon) = \{(s, h, h) \mid h \in Q\}$$
$$\delta'(s', a) = \varnothing$$

$$\delta'((p, h, q), \varepsilon) = \varnothing$$
$$\delta'((p, h, q), a) = \big\{\big(\delta(p, a), h, \delta(q, a)\big)\big\}$$

M' reads its input string w and simulates M reading the input string ww. Specifically, M' simultaneously simulates two copies of M, one reading the left half of ww starting at the usual start state s, and the other reading the right half of ww starting at some intermediate state h.

- The new start state s' non-deterministically guesses the "halfway" state $h = \delta^*(s, w)$ without reading any input; this is the only non-determinism in M'.
- State (p, h, q) means the following:
  - The left copy of M (which started at state s) is now in state p.
  - The initial guess for the halfway state is h.
  - The right copy of M (which started at state h) is now in state q.
- M' accepts if and only if the left copy of M ends at state h (so the initial non-deterministic guess $h = \delta^*(s, w)$ was correct) and the right copy of M ends in an accepting state.

■

Solution (smartass): A complete solution is given in the lecture notes. ■

Rubric: 5 points: standard langage transformation rubric (scaled). Yes, the smartass solution would be worth full credit.

(b) Describe a regular language L such that the language $double(L) := \{ww \mid w \in L\}$ is *not* regular. Prove your answer is correct.

Solution: Consider the regular language $L = 0^*1$.

Expanding the regular expression lets us rewrite $L = \{0^n1 \mid n \geq 0\}$. It follows that $double(L) = \{0^n10^n1 \mid n \geq 0\}$. I claim that this language is not regular.

Let x and y be arbitrary distinct strings in L.

Then $x = 0^i1$ and $y = 0^j1$ for some integers $i \neq j$.

Then x is a distinguishing suffix of these two strings, because

- $xx \in double(L)$ by definition, but
- $yx = 0^i10^j1 \notin double(L)$ because $i \neq j$.

We conclude that L is a fooling set for $double(L)$.

Because L is infinite, $double(L)$ cannot be regular. ■

Solution: Consider the regular language $L = \Sigma^* = (0 + 1)^*$.

I claim that the language $double(\Sigma^*) = \{ww \mid w \in \Sigma^*\}$ is not regular.

Let F be the infinite language 01^*0.

Let x and y be arbitrary distinct strings in F.

Then $x = 01^i0$ and $y = 01^j0$ for some integers $i \neq j$.

The string $z = 1^i$ is a distinguishing suffix of these two strings, because

- $xz = 01^i01^i = ww$ where $w = 01^i$, so $xz \in double(\Sigma^*)$, but
- $yx = 01^j01^i \notin double(\Sigma^*)$ because $i \neq j$.

We conclude that F is a fooling set for $double(\Sigma^*)$.

Because F is infinite, $double(\Sigma^*)$ cannot be regular. ■

Rubric: 5 points:

- 2 points for describing a regular language L such that *double*(L) is not regular.
- 1 point for describing an infinite fooling set for *double*(L):
  - + ½ for explicitly describing the proposed fooling set F.
  - + ½ if the proposed set F is actually a fooling set.
  - — No credit for the proof if the proposed set is not a fooling set.
  - — No credit for the *problem* if the proposed set is finite.
- 2 points for the proof:
  - + ½ for correctly considering *arbitrary* strings x and y
    - — No credit for the proof unless both x and y are *always* in F.
    - — No credit for the proof unless both x and y can be *any* string in F.
  - + ½ for correctly stating a suffix z that distinguishes x and y.
  - + ½ for proving either $xz \in L$ or $yz \in L$.
  - + ½ for proving either $yz \notin L$ or $xz \notin L$, respectively.

These are not the only correct solutions. These are not the only fooling sets for these languages.

Standard langage transformation rubric. For problems worth 10 points:

+ 2 for a formal, complete, and unambiguous description of the output automaton, including the states, the start state, the accepting states, and the transition function, as functions of an *arbitrary* input DFA. The description must state whether the output automaton is a DFA, an NFA without ε-transitions, or an NFA with ε-transitions.
  - No points for the rest of the problem if this is missing.
+ 2 for a *brief* English explanation of the output automaton. We explicitly do *not* want a formal proof of correctness, or an English *transcription*, but a few sentences explaining how your machine works and justifying its correctness. What is the overall idea? What do the states represent? What is the transition function doing? Why these accepting states?
  - **Deadly Sin:** No points for the rest of the problem if this is missing.
+ 6 for correctness
  + 3 for accepting *all* strings in the target language
  + 3 for accepting *only* strings in the target language
  − 1 for a single mistake in the formal description (for example a typo)
  - Double-check correctness when the input language is $\varnothing$, or $\{\varepsilon\}$, or $\mathtt{0}^*$, or Σ^*.