Final #2

Mark all correct answers for each of the following questions.

Σ denotes an arbitrary fixed alphabet and L an arbitrary language over Σ, unless otherwise specified.

1. Given a language L over the alphabet $\{a, b\}$, denote:

$$L' = \{w \in L : abw = wab\}.$$

   (a) If $L_1, L_2 \subseteq \{a, b\}^*$, then $(L_1 \cup L_2)' = L_1' \cup L_2'$.

   (b) If $L_1, L_2 \subseteq \{a, b\}^*$, then $(L_1 \cap L_2)' = L_1' \cap L_2'$.

   (c) If $L_1, L_2 \subseteq \{a, b\}^*$, then

$$(L_1 L_2)' = L_1' L_2' \cup (L_1 \cap \{ab\}^*\{a\})\,(L_2 \cap \{b\}\{ab\}^*).$$

   (d) If $L \subseteq \{a, b\}^*$, then $(L^*)' = (L')^*$.

   (e) If there exists a finite automaton accepting L, then there exists a finite automaton accepting L'.

   (f) The existence of a pushdown automaton accepting L does not necessarily imply the existence of a pushdown automaton accepting L'.

   (g) None of the above.

2. (a) The sequence $(b_n)_{n=0}^{\infty}$ is defined by:

$$b_n = n \bmod 10, \qquad n = 0, 1, 2, \ldots .$$

   The sequence $(a_n)_{n=0}^{\infty}$ is defined by the recursion

$$a_{n+1} = 10a_n + b_n, \qquad n = 0, 1, 2, \ldots ,$$

   and the initial condition $a_0 = 0$. Then the language consisting of the base 10 expansions of all the a_n's is regular.

(b) For each non-negative integer n, let $t_n \in \{0,1,\ldots,9\}^*$ be the base 10 expansion of n. The language

$$L = \{t_n^n : n \geq 0\} = \{\varepsilon, 1, 22, 333, 4444, \ldots\}$$

is regular.

(c) The language $L \subseteq \Sigma^*$ is known to have the property that for every $u, v, w \in \Sigma^*$ there exists an $n \geq 0$ for which $uv^n w \in L$. Then L is regular.

(d) The language $L \subseteq \Sigma^*$ is known to have the property that for every $u, v, w \in \Sigma^*$ with $v \neq \varepsilon$ there exists an $n \geq 0$ for which $uv^n w \in L$. Then L is regular.

(e) If G is given by the rules
$S \to \varepsilon \mid aSb \mid SS$,
then every word $w \in L(G)$ is yielded by infinitely many distinct parse trees over G.

(f) Let G be a context-free grammar for which there exists a word $w \in L(G)$ yielded by two distinct parse trees over G. Then the word w is yielded by infinitely many distinct parse trees over G.

(g) None of the above.

3. Given a sequence $(L_n)_{n=1}^{\infty}$ of languages and a language L over Σ, we say that L_n *converges to* L, and denote $L_n \underset{n\to\infty}{\longrightarrow} L$, if for every $w \in L$ we have $w \in L_n$ for all sufficiently large n and for every $w \notin L$ we have $w \notin L_n$ for all sufficiently large n.

(a) Let $L_n \underset{n\to\infty}{\longrightarrow} L$. If all L_n's are regular then L is not necessarily regular, and if all L_n's are context-free then L is not necessarily context-free. However, if all L_n's are Turing-accepted then L is Turing-accepted

(b) If $L_n \underset{n\to\infty}{\longrightarrow} L$ and L is Turing-accepted, then L_n is Turing-accepted for all sufficiently large n.

(c) If $G = (N, \Sigma, R, S)$ is a context-free grammar, then for every $v \in (N \cup \Sigma)^*$ the language $\left\{w \in \Sigma^* : v \overset{*}{\underset{G}{\Longrightarrow}} w\right\}$ is context-free.

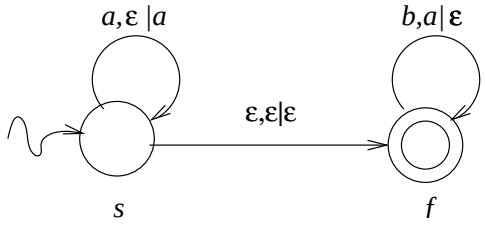
(d) If $M = (Q, \Sigma, \Gamma, \Delta, s, A)$ is a pushdown automaton, then for every $p, q \in Q$ and $\alpha, \beta \in \Gamma^*$ the language $\left\{w \in \Sigma^* : (p, w, \alpha) \vdash_M^* (q, \varepsilon, \beta)\right\}$ is context-free.

(e) If $M = (Q, \Sigma, \Gamma, \Delta, s, A)$ is a pushdown automaton satisfying $|\alpha| \geq |\beta|$ for every $((p, w, \alpha), (q, \beta)) \in \Delta$, then $L(M)$ is regular.

(f) If $M = (Q, \Sigma, \Gamma, \Delta, s, A)$ is a pushdown automaton satisfying $|\alpha| \leq |\beta|$ for every $((p, w, \alpha), (q, \beta)) \in \Delta$, then $L(M)$ is regular.

(g) None of the above.

4. Let M be the pushdown automaton described by the following diagram:



For every $k \geq 0$ let M_k be the pushdown automaton obtained from M by adding to it the transition $((f, c, a^k), (s, a^k))$. For $k, l \geq 0$ let $M_{k,l}$ be the automaton obtained from M by adding to it the transition $((f, c^l, a^k), (s, a^k))$. (Thus $M_k = M_{k,1}$.)

(a) All languages $L(M_k)$ contain properly the language $L(M)$. Moreover, we have $L(M_0) \supseteq L(M_1) \supseteq L(M_2) \supseteq \ldots$. However, since all languages are countable, it is impossible for all these languages to be distinct, and therefore $L(M_{k+1}) = L(M_k)$ for all sufficiently large k.

(b) Let M' be the simple pushdown automaton, equivalent to M, constructed according to the algorithm presented in class. Let M'_k be the automaton obtained from M' by adding to it the transition $((f, c, a^k), (s, a^k))$. Then the set of automata $\{M'_k : k \geq 0\}$ contains exactly two simple automata.

(c) Let $G_{10} = (N_{10}, \{a, b, c\}, R_{10}, S)$ be the context-free grammar satisfying $L(G_{10}) = L(M_{10})$, constructed according to the algorithm presented in class. Then $|N_{10}| = 289$.

(d) The set R_{10} of rules in the grammar of the preceding part consists of more than 1000000 rules.

(e) If M_0 is changed by adjoining the state s to the set of accepting states, then the language accepted by the automaton does not change.

(f) It is possible to find an infinite sequence (k_1, l_1), (k_2, l_2), ... of pairs of non-negative integers such that for every $i \neq j$ neither of the two languages $L\left(M_{k_i,l_i}\right)$ and $L\left(M_{k_j,l_j}\right)$ contains the other.

(g) None of the above.

5. Given a Turing machine $M = (Q, \Sigma, \Gamma, \delta, s, h)$, denote by $C = \Gamma^* \times Q \times (\Gamma^*(\Gamma - \{B\}) \cup \{\varepsilon\})$ the space of all configurations over M. (Recall that we denote a typical configuration by uqv rather than (u, q, v), but this is of no consequence.)

(a) If M is a Turing machine, then there cannot exist an input word $w \in \Sigma^*$ such that $sw \overset{*}{\underset{M}{\vdash}} c$ for every $c \in C$.

(b) If the set $\left\{c \in C : s \overset{*}{\underset{M}{\vdash}} c\right\}$ is infinite, then M may decide a finite language, but cannot possibly decide an infinite language.

(c) If there exists a $\sigma \in \Sigma$ such that $s\sigma \overset{*}{\underset{M}{\vdash}} s$, then M does not decide a language.

(d) If M computes a function $f : \Sigma^* \longrightarrow \Gamma^*$, and for every two input words $w_1, w_2 \in \Sigma^*$ with $w_1 \neq w_2$ we have $\left\{c \in C : sw_1 \overset{*}{\underset{M}{\vdash}} c\right\} \cap \left\{c \in C : sw_2 \overset{*}{\underset{M}{\vdash}} c\right\} = \emptyset$, then f is one-to-one.

(e) If M computes a function $f : \Sigma^* \longrightarrow \Gamma^*$, and there exist two input words $w_1, w_2 \in \Sigma^*$ with $w_1 \neq w_2$ such that $\left\{c \in C : sw_1 \overset{*}{\underset{M}{\vdash}} c\right\} \cap \left\{c \in C : sw_2 \overset{*}{\underset{M}{\vdash}} c\right\} \neq \emptyset$, then f is not one-to-one.

(f) If M_1 computes a constant function f (that is, $f(w)$ is the same for all $w \in \Sigma^*$) and $M_1 M_2$ decides a language L, then either $L = \Sigma^*$ or $L = \emptyset$.

(g) None of the above.

Solutions

1. Two words commute if and only if they are both powers of a third word. In particular, we have $abw = wab$ if and only if $w = (ab)^k$ for some $k \geq 0$. Hence $L' = L \cap L((ab)^*)$. This implies

$$\begin{aligned}(L_1 \cup L_2)' &= (L_1 \cup L_2) \cap L((ab)^*) \\ &= (L_1 \cap L((ab)^*)) \cup (L_2 \cap L((ab)^*)) = L_1' \cup L_2'\end{aligned}$$

and

$$\begin{aligned}(L_1 \cap L_2)' &= (L_1 \cap L_2) \cap L((ab)^*) \\ &= (L_1 \cap L((ab)^*)) \cap (L_2 \cap L((ab)^*)) = L_1' \cap L_2'\,.\end{aligned}$$

Now:

$$(L_1 L_2)' = L_1 L_2 \cap L((ab)^*)\,.$$

A word $w_1 w_2$ is of the form $(ab)^k$ if and only if either both w_1 and w_2 are of this form or $w_1 = (ab)^l a$ and $w_2 = b(ab)^m$ for some $l, m \geq 0$. Consequently:

$$(L_1 L_2)' = L_1' L_2' \cup (L_1 \cap \{ab\}^*\{a\})(L_2 \cap \{b\}\{ab\}^*)\,.$$

If $L = \{a, b\}$, then $(L^*)' = L((ab)^*)$, but $(L')^* = \emptyset^* = \{\varepsilon\}$.

The closure properties of the family of regular languages guarantee that, since $L((ab)^*)$ is regular, so is L' for any regular L. Similarly, L' is context-free for any context-free L.

Thus, (a), (b) and (e) are true.

2. Since each b_n is a 1-digit number, the decimal expansion of a_{n+1} is the concatenation of the decimal expansion of a_n and the digit defining b_n. Now b_n attains periodically the values $0, 1, \ldots, 9$, and therefore the language consisting of the decimal expansions of all the a_n's is

$$\{0\} \cup \{1234567890\}^*\{1, 12, 123, \ldots, 123456789\}\,,$$

which is regular.

Obviously, $|t_n^n| = n|t_n|$. Since the numbers $|t_n|$ are non-decreasing, the numbers $n|t_n|$ are strictly increasing. Moreover, for $n = 10^k$ we have

$$n|t_n| - (n-1)|t_{n-1}| = 10^k(k+1) - \left(10^k - 1\right)k = 10^k + k.$$

Thus the set of all wordlengths of the language $\{t_n^n : n \geq 0\}$ is of unbounded gaps, so that the language is not regular (and not even context-free).

If for every $u, v, w \in \Sigma^*$ there exists an $n \geq 0$ for which $uv^n w \in L$, then, by taking $u = v = \varepsilon$, we obtain $w \in L$ for every $w \in \Sigma^*$, so that $L = \Sigma^*$. Now suppose that the condition is only assumed to hold for triplets of words u, v, w with $v \neq \varepsilon$. Order in some way all triplets of such words: (u_1, v_1, w_1), $(u_2, v_2, w_2), \ldots$. Let $(a_n)_{n=1}^{\infty}$ be a sequence of non-negative integers, to be determined later. The language $L = \{u_k v_k^{a_k} w_k : k \geq 1\}$ satisfies the condition in question. By taking the sequence(a_n) to increase sufficiently fast, we may ensure that the sequence $(|u_k| + a_k|v_k| + |w_k|)_{k=1}^{\infty}$ increases arbitrarily fast and consequently the set of all wordlengths of the language L is of unbounded gaps, so that L is not regular.

Let G be given by the rules
$S \to \varepsilon \mid aSb \mid SS$,
and $w \in L(G)$. For any parse tree yielding w and corresponding sequence of derivations leading from S to w, we may take the same sequence of derivations, preceded by the two derivations $S \Longrightarrow SS \Longrightarrow S$, and obtain another parse tree yielding w. Continuing this process, we obtain infinitely many such parse trees. On the other hand, if G is given by the rules

- $S \to A \mid \varepsilon$,
- $A \to \varepsilon$,

then there are exactly two parse trees yielding ε.

Thus, (a), (c) and (e) are true.

3. For any language L there exists a sequence $(L_n)_{n=1}^{\infty}$ of regular languages converging to it. In fact, if L is finite, we can just take $L_n = L$

for each n. If L is infinite, say $L = \{w_1, w_2, \ldots\}$, then take $L_n = \{w_1, w_2, \ldots, w_n\}$ for each n. Obviously, in both cases the L_n's are regular (being finite) and converge to L. Thus, the fact that there exists a sequence of regular languages converging to some language gives no information regarding this language.

Similarly, the fact that a sequence of languages converges to a given language provides little information about the properties of the languages in the sequence. For example, let $L = \Sigma^*$ and let L_0 be any language which is not Turing-accepted. Write $\Sigma^* - L_0 = \{w_1, w_2, \ldots\}$. Put $L_n = L_0 \cup \{w_1, w_2, \ldots, w_n\}$ for each n. Each L_n is not Turing-accepted (otherwise the language L_0, obtained from it by omitting finitely many words, would be Turing-accepted as well), yet $L_n \underset{n\to\infty}{\longrightarrow} L$.

Given a context-free grammar $G = (N, \Sigma, R, S)$ and $v \in (N \cup \Sigma)^*$, let $G_1 = (N \cup \{S_1\}, \Sigma, R \cup \{(S_1, v)\}, S_1)$, where $S_1 \notin N$. Then for $w \in (N \cup \Sigma)^*$ we have $v \underset{G}{\overset{*}{\Longrightarrow}} w$ if and only if $S_1 \underset{G_1}{\overset{*}{\Longrightarrow}} w$. In particular, $\left\{w \in \Sigma^* : v \underset{G}{\overset{*}{\Longrightarrow}} w\right\} = L(G_1)$, so that the left hand side is a context-free language.

Given a pushdown automaton $M = (Q, \Sigma, \Gamma, \Delta, s, A)$, states $p, q \in Q$ and words $\alpha, \beta \in \Gamma^*$, construct a pushdown automaton

$$M_1 = (Q \cup \{p_0, q_0\}, \Sigma, \Gamma, \Delta_1, p_0, \{q_0\}),$$

where $p_0, q_0 \notin Q$ and

$$\Delta_1 = \Delta \cup \{((p_0, \varepsilon, \varepsilon), (p, \alpha))\} \cup \{((q, \varepsilon, \beta), (q_0, \varepsilon))\} .$$

Obviously, $(p, w, \alpha) \underset{M}{\overset{*}{\vdash}} (q, \varepsilon, \beta)$ if and only if $((p_0, w, \varepsilon) \underset{M_1}{\overset{*}{\vdash}} (q_0, \varepsilon, \varepsilon)$, namely if and only if $w \in L(M_1)$. Hence the language

$$\left\{w \in \Sigma^* : (p, w, \alpha) \underset{M}{\overset{*}{\vdash}} (q, \varepsilon, \beta)\right\}$$

is context-free.

Let w be accepted by a pushdown automaton $M = (Q, \Sigma, \Gamma, \Delta, s, A)$. Consider a sequence of configurations leading from (s, w, ε) to $(f, \varepsilon, \varepsilon)$, where $f \in A$. If any of the intermediate configurations has a non-empty word in the stack, then the first transition leading to such a

configuration must be of the form $((p, u, \varepsilon), (q, \beta_0))$ for some $\beta_0 \neq \varepsilon$. In this case we must also have some last configuration with a non-empty stack. The transition leading from this configuration must be of the form $((p, u, \alpha_0), (q, \varepsilon))$ for some $\alpha_0 \neq \varepsilon$. Hence, if either $|\alpha| \geq |\beta|$ for every $((p, w, \alpha), (q, \beta)) \in \Delta$, or $|\alpha| \leq |\beta|$ for every $((p, w, \alpha), (q, \beta)) \in \Delta$, words in $L(M)$ will be generated only by transitions of the form $((p, u, \varepsilon), (q, \varepsilon))$. But then we have effectively an NFA, so that the accepted language is regular.

Thus, (c), (d), (e) and (f) are true.

4. For each k we have $L(M_{k+1}) \subseteq L(M_k)$. In fact, the sequence of transitions showing that a word belongs to $L(M_k)$ shows that the word belongs to $L(M_{k+1})$ as well. On the other hand, $a^k c b^k \in L(M_k) - L(M_{k+1})$, so that the above inclusions are all proper.

   M'_k is simple if and only if the transition $((f, c, a^k), (s, a^k))$ does not prevent it from being such. For $k = 0$ it is not simple as it should contain the transition $((f, c, a), (s, a))$ once it contains $((f, c, \varepsilon), (s, \varepsilon))$. For $k \geq 2$ it is not simple because the transition involves a removal of a word of length greater than 1 from the stack. Thus M'_k is simple only for $k = 1$.

   The simple automaton equivalent to M_{10} is obtained from M by first adding the two transitions $((s, a, a), (s, a^2))$ and $((s, \varepsilon, a), (f, a))$, and then letting the transition $((f, c, a^{10}), (s, a^{10}))$ be carried out in 10 steps instead of 1. Namely, we have to add 9 new states $p_1, p_2, \ldots, p_9$ and transitions

   $$((f, \varepsilon, a), (p_1, \varepsilon)), ((p_0, \varepsilon, a), (p_1, \varepsilon)), \ldots, ((p_8, \varepsilon, a), (p_9, \varepsilon)), ((p_9, \varepsilon, a), (s, a^{10})).$$

   The resulting automaton has a state set of size 11 and a stack alphabet of size 1, and by the construction we have $|N_{10}| = 1 + 11 \cdot (1 + 1) \cdot 11 = 243$. The transition $((p_9, \varepsilon, a), (s, a^{10}))$ yields by itself the 11^{10} type-2 grammatical rules
   $\langle p_9, a, q\rangle \to \langle s, a, q_1\rangle\langle q_1, a, q_2\rangle \ldots \langle q_9, a, q\rangle$
   where $q, q_1, \ldots, q_9$ are any states in $\{s, f, p_1, \ldots, p_9\}$, so that $|R_{10}|$ is much larger than 1000000.

In general, if an automaton is changed by turning some non-accepting state q into an accepting state, the language accepted by the new automaton certainly contains that accepted by the old one. Clearly, however, if the set of transitions includes a transition of the form $((q, \varepsilon, \varepsilon), (f, \varepsilon))$ for some accepting state f, then the change does not add any new words to the language. Hence changing M_0 by making s an accepting state does not change the language accepted by the automaton.

Consider the language $L(M_{1,p})$ for any prime p. Obviously, the number of occurrences of c in any word in this language is divisible by p. Also, the word $ac^p b$ belongs to this language. It follows that, for any two distinct primes p_1 and p_2, neither of the languages $L(M_{1,p_1})$ and $L(M_{1,p_2})$ contains the other.

Thus, (d), (e) and (f) are true.

5. Since a Turing machine halts upon arriving at the state h, in the course of any computation the machine may get to at most one configuration of the form uhv with $u, v \in \Gamma^*$.

   If the set $\left\{c \in C : s \overset{*}{\underset{M}{\vdash}} c\right\}$ is infinite, then M does not halt for input ε, and therefore cannot possibly decide any language.

   Consider a Turing machine operating as follows. At the first step, if the leftmost square on the tape is blank, then the machine writes Ⓨ and halts. If the leftmost square is not blank, then the machine erases the tape and then moves to the state s. Then, as before, it writes Ⓨ at the leftmost square and halts. Thus, we have $sw \overset{*}{\underset{M}{\vdash}} s$ for every $w \in \Sigma^*$, yet M decides Σ^*.

   If M computes a function $f : \Sigma^* \longrightarrow \Gamma^*$, and

$$\left\{c \in C : sw_1 \overset{*}{\underset{M}{\vdash}} c\right\} \cap \left\{c \in C : sw_2 \overset{*}{\underset{M}{\vdash}} c\right\} \neq \emptyset$$

   for some $w_1 \neq w_2$, then the computation yields the same results for inputs w_1 and w_2. Thus $f(w_1) = f(w_2)$, so that f is not one-to-one. However, knowing that distinct inputs never lead to the same configuration does not imply that they yield distinct outputs, as the configuration takes into account also the head's position, which is basically

immaterial at the stage when the machine halts. For example, suppose a machine halts right away unless the head points at a blank square, in which case the machine writes some letter σ_0 on the tape, moves the head to the right and halts. Then the machine computes the function $f : \Sigma^* \longrightarrow \Sigma^*$ given by

$$f(w) = \begin{cases} w, & w \neq \varepsilon, \\ \sigma_0, & w = \varepsilon. \end{cases}$$

The function is not one-to-one since $f(\sigma_0) = f(\varepsilon)$. However, for input σ_0 we arrive at the configuration $h\sigma_0$, whereas for input ε we arrive at the configuration $\sigma_0 h$.

If M_1 computes a constant function f, then $M_1 M_2$ may still decide a non-trivial language. This may be the case if the head is not at the same location for all inputs. For example, suppose $sw \overset{*}{\underset{M}{\vdash}} h\gamma_0$ if $w \in L(a^*)$, while $sw \overset{*}{\underset{M}{\vdash}} \gamma_0 h$ otherwise. It is then easy to design M_2 so that $M_1 M_2$ will decide the language $L(a^*)$.

Thus, only (a) and (e) are true.