1. ***Regular or not?*** *Prove or disprove* that each of the languages below is regular (or not). Let Σ^+ denote the set of all *nonempty* strings over alphabet Σ; in other words, $\Sigma^+ = \Sigma \cdot \Sigma^*$. Denote $n(w)$ the integer corresponding to the binary string w.

(a) $\{\texttt{3}x\texttt{=}y : x, y \in \{\texttt{0}, \texttt{1}\}^*, n(y) = 3n(x)\}$

Solution: Denote the language in the problem 1(a) as L_a. We prove that L_a is not regular by constructing a fooling set for L_a of infinite size.

Let $F = \{\texttt{310}^i : i \geq 0\}$. For two distinct prefixes $x = \texttt{310}^i$ and $y = \texttt{310}^j$ in F, let z be $\texttt{=110}^i$.

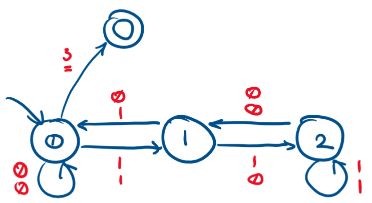
- $xz = \texttt{310}^i\texttt{=110}^i$; because $n(\texttt{110}^i) = 3n(\texttt{10}^i)$, we have xz in F.
- $yz = \texttt{310}^j\texttt{=110}^i$; because $n(\texttt{110}^i) \neq 3n(\texttt{10}^j)$ if $i \neq j$, we have yz not in F.

This implies that F is a fooling set of infinite size, and thus L_a is not regular. ■

(b) $\left\{ \substack{\texttt{3x}\\ \texttt{=y}} : \substack{\texttt{x}\\ \texttt{y}} \in \left\{ \substack{\texttt{0}\\ \texttt{0}}, \substack{\texttt{0}\\ \texttt{1}}, \substack{\texttt{1}\\ \texttt{0}}, \substack{\texttt{1}\\ \texttt{1}} \right\}^*, n(y) = 3n(x) \right\}$

Solution: Denote the language in the problem 1(b) as L_b. We prove that L_b is regular by constructing an NFA recognizing L_b.

We construct NFA recognizing the reverse of the language, L_b^R; by the exercise problems, L_b is regular if and only if L_b^R is regular.



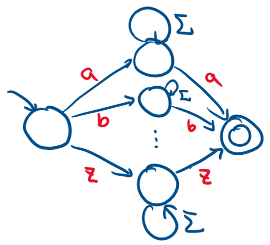
The NFA reads the input from the least significant bits of x and y, and records the amount of carry at any moment. The transitions are implemented so that the machine only continues if the current digit of y equals to (the least significant bit of) three times the corresponding digit in the x plus the carry. After reading the full strings x and y, if there is any carry left then we reject; otherwise the NFA finishes off by reading the leading $\substack{\texttt{3}\\ \texttt{=}}$ and accepts. ■

(c) $\{wxw^R : w, x \in \Sigma^+\}$

Solution: Denote the language in the problem 1(c) as L_c. We prove that L_c is regular by constructing an NFA recognizing L_c, which is equivalent to the following language:

$$L_c' := \{\sigma x' \sigma : x' \in \Sigma^+, \sigma \in \Sigma\}.$$

For $L_c' \subseteq L_c$, take $w = \sigma$ and $x = x'$; for $L_c \subseteq L_c'$, take σ to be the first symbol in w and x' to be whatever is left.



The constructed NFA reads the first and the last symbol, and accepts if they match; therefore the NFA correctly recognizes language L_c'. More formally, create one state q_σ for each symbol

$\sigma \in \Sigma$; and add two extra states s and t. Let s be the only starting state and t be the only accepting state. For each symbol σ, add transitions s to q_σ and q_σ to t on reading σ, and self-loop transition at q_σ on reading all symbols.

■

(d) $\{ww^Rx : w, x \in \Sigma^+\}$

Solution: Denote the language in the problem 1(d) as L_d. We prove that L_d is not regular by constructing a fooling set for L_d of infinite size. Without loss of generality we assume that `0` and `1` are in Σ.

Let $F = \{01^i0 : i \text{ is an odd integer}\}$. For two distinct prefixes $u = 01^i0$ and $v = 01^j0$ in F (without loss of generality assuming $i < j$), consider the suffix $z = 01^i00$.

- $uz = 01^i001^i00$; by taking $w = 01^i0$ and $x = 0$, this shows that uz is in F.
- $vz = 01^j001^i00$. Because j is odd, ww^R cannot be of the form 01^j0; which means the first run of `1`s must lie in w completely. But then there are not enough `1`s in the rest of the word to form w^R. Therefore, not matter what x is, word vz cannot be of the form ww^Rx. This shows that vz is not in F.

This implies that F is a fooling set of infinite size, and thus L_d is not regular. ■

Rubric: Standard 5-point grading scale (plus deadly-sins and sudden-death rules) *for each subproblem*, scaled to 2.5 points. (Thus the total 10 points for problem 1.) Maximum 1.5 points if one tries to prove a regular language to be non-regular, or vice versa. Maximum 0.5 point if both guesses are wrong. Maximum 0.5 points if the fooling set is in fact not fooling.

Full credit for subproblem (d) if one correctly proves the language to be regular *when Σ is unary*. This was an oversight.

2. ***Telling DFAs apart.***

Let M_1 and M_2 be two DFAs, each with exactly n states. Assume that the languages associated with the two machines are different (that is, $L(M_1) \neq L(M_2)$), there is always some string in the symmetric difference of the two languages.

Prove that there is a string w of length polynomial in n in the symmetric difference of $L(M_1)$ and $L(M_2)$. What is the best upper bound you can get on the length of w?

Solution: First we construct a DFA M', described by $(Q', s', A', \Sigma', \delta')$, that recognizes the symmetric difference of the two languages $L(M_1)$ and $L(M_2)$, using the product construction. Denote M_i by the tuple $(Q_i, s_i, A_i, \Sigma_i, \delta_i)$ for $i \in \{1, 2\}$.

- States Q': $Q_1 \times Q_2$ — pairs of states, one from each M_i
- Starting state s': (s_1, s_2)
- Accepting states A': $\{(r_1, r_2) \in Q' : \text{either } r_1 \in A_1 \text{ and } r_2 \notin A_2, \text{ or } r_1 \notin A_1 \text{ and } r_2 \in A_2\}$
- Alphabet Σ': $\Sigma_1 \cup \Sigma_2$
- Transition function δ': $\delta_1 \times \delta_2$, mapping $\delta'((q_1, q_2), \mathtt{a})$ to $(\delta_1(q_1, \mathtt{a}), \delta(q_2, \mathtt{a}))$ on reading any symbol $\mathtt{a} \in \Sigma'$

DFA M' recognizes the symmetric difference of the two languages $L(M_1)$ and $L(M_2)$ and has n^2 states. Now by problem statement M' accepts at least one word. Now any walk from the starting state s' to an accepting state in M' can be turned into a simple path between the same two endpoints, without ever visiting the same state twice. This shows that there is a word of length at most n^2 that is accepted by M', and thus in the symmetric difference of $L(M_1)$ and $L(M_2)$. ■

Rubric: Any complete solution with a (justified) polynomial bound receives full credit. Any *subquadratic* bound receives extra credit.

Standard 5-point grading scale plus deadly-sins and sudden-death rules.