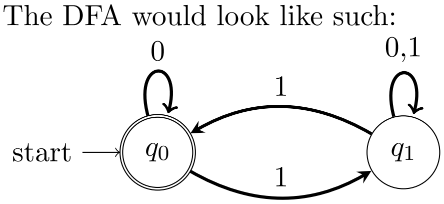
COMS 331: Theory of Computing, Spring 2023
Homework Assignment 3

Neha Maddali
Due at 10:00PM, Wednesday, February 15, on Gradescope.

Problem 16. Construct a 2-state DFA that decides the language $A = \{x \in \{0,1\}^* |$ bnum(x) is divisible by 3$\}$. The 2 states must represent "divisible by 3" and "not divisible by 3." "Divisible by 3" is an accepting state and "not divisible by 3" is not an accepting state. The transition function is described as such:

- adding a 0 to the end of a binary number doubles the number, which the result will always be divisible by 3
- adding a 1 to the end of a binary number divisible by 3 results in a binary number not divisible by 3
- adding a 0 to the end of a binary number not divisible by 3 results in a binary number not divisible by 3
- adding a 1 to the end of a binary number not divisible by 3 results in either a binary number divisible by 3 or not divisible by 3. For example, $10_2 = 2_{10}$ and $101_2 = 5_{10}$ and neither are divisible by 3. $111_2 = 7_{10}$ and $1111_2 = 15_{10}$. 7 isn't divisible by 3 but 15 is.

The DFA would look like such:

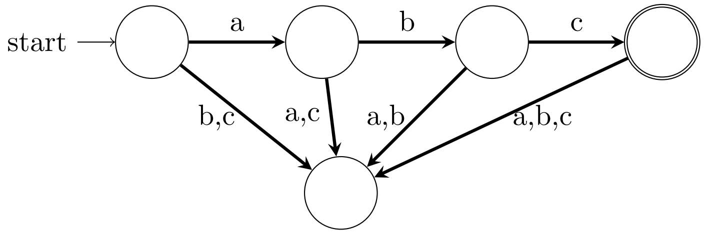


There is a contradiction because this diagram is not a DFA. IT is a nondeterministic finite automaton because state q_1 has 2 different transitions if the bit is a 1. Thus, no 2-state DFA can decide A. ■

Problem 17. A language $A \subseteq \Sigma^*$ is regular if there exists a DFA M such that L(M) = A. So to prove that the singleton language $\{x\}$ is regular for all $x \in \Sigma^*$, we must show that there is a DFA M such that L(M)=$\{x\}$. We know that $\{x\}$ is a finite language because it consists of only one string and that one string is finite because all strings in Σ^* are finite. Thus, we know that a DFA M exists such that L(M)=$\{x\}$.

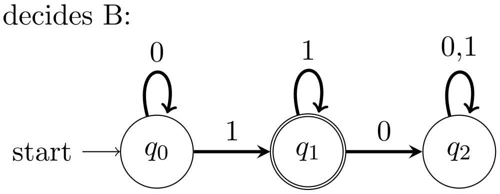
The DFA would have $|x| + 2$ states: one for the start state, one for a trap state, and one for each character in x. Starting at the start state, the transition function would transition to the next non-trap state following the path that matches the first character in x. All other characters transition to a trap state. This repeats with the path of the second character in x up until the last character, in which the last state is the only accepting state. Once in the accepting state, any other character that follows will transition to the trap state.

An example DFA for the language $\{abc\}$ is:



Problem 18. Each finite language is made up of singleton languages, whether it be a singleton language itself or the union of a finite amount of singleton languages. Essentially we can use what we learned in lecture about product construction to write this proof since we also learned that it works for union. So let $M_A = (Q_A, \Sigma, \delta_A, s_A, F_A)$ and $M_B = (Q_B, \Sigma, \delta_B, s_B, F_B)$ then the union DFA is defined as $M = (Q_A \cup Q_B, \Sigma, \delta, s, F)$ where $\delta : (Q_A \cup Q_B) \times \Sigma = (Q_A \cup Q_B)$ is defined by $\delta((q, r), a) = \delta_A(q, a), \delta_B(r, a)$ for all $q \in Q_A, r \in Q_B$, and $a \in \Sigma$. Also $s = (s_A, s_B)$ and $F = \{(q, r) \in Q_A \cup Q_B | q \in F_A \text{ or } q \in F_B\}$. So now, we can construct our DFA that is a union of all singleton languages in our finite language, and we have a finite amount of singleton languages, which thus proves that every finite language $A \subseteq \Sigma^*$ is regular. ■

Problem 19. Let language $B = \{0^n 1^m | n, m \in N\}$. B is regular because there exists a DFA that decides B:

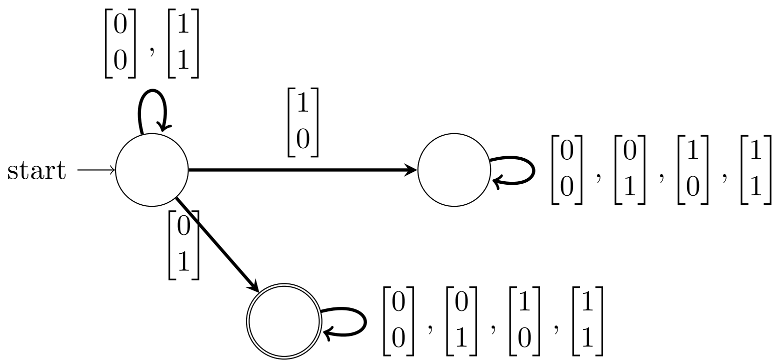


Then according to the claim that every subset of a regular language is regular, every subset of B must also be a regular language. Let language $A = \{0^n 1^n | n \in N\}$. It is true that $A \subseteq B$ based on the definitions of languages. But we know that A is not a regular language as proved in lecture.

To give a summary of the proof from lecture, there exists some $0 \leq i \leq j \leq k$ where k is the number of states in the DFA such that $\hat{\delta}(s, 0^i) = \hat{\delta}(s, 0^j)$ because there are k+1 items in the list $\hat{\delta}(s, \epsilon), \hat{\delta}(s, 0), \hat{\delta}(s, 00)..., \hat{\delta}(s, 0^k)$ but only k states. Then $0^j 1^i$ and $0^i 1^i$ would lead to the same state, so the DFA has to either accept both strings or reject both strings. However, $0^i 1^i$ is in the set while $0^j 1^i$ is not, so there is a contradiction.

Because $A \subseteq B$, A is not regular and B is regular, there is a contradiction to the original claim. Not every subset of a regular language must be regular. ■

Problem 20. DFA:



Problem 21. Say we have $\Sigma = \{0, 1\}$ and language $B = \{0^n 1^n | n \in N\}$. B can be constructed by the union of singleton languages $A \subseteq \Sigma^*$, where $A_0 = \{\epsilon\}, A_1 = \{01\}, A_2 = \{0011\}$.... We proved in problem 17 that for all $x \in \Sigma^*$, the singleton language $\{x\}$ is regular. Thus, $B = \bigcup_{n=0}^{\infty} A_n$ will be regular. However, there is a contradiction because we know that B is not regular according to lecture and also summarized in problem 19. Therefore, the original claim is false and $A_n \subseteq \Sigma^*$ being regular for each $n \in N$ doesn't necessarily mean that $\bigcup_{n=0}^{\infty} A_n$ will be regular. ■

Problem 22.