CSE 103 Computational Models

Fall 2021 Prof. Fremont **HW 1**

(5 questions, 215 points total)

Note: Some problems may have multiple correct solutions: there are often multiple ways to prove a statement, or to construct a DFA, for example. We can't give all possible correct answers here, but we give at least one; you won't lose any points if your answer is different from ours but still correct.

1. (25 pts.) Alphabets and Strings

Let $\Sigma = \{a,b,c\}$. Recall that we denote the empty string by ε.

(a) What is the set Σ^2?

Solution: All strings of length 2 over Σ: $\{aa, ab, ac, ba, bb, bc, ca, cb, cc\}$.

(b) What is $|\Sigma^5|$?

Solution: For each element of a string of length 5 over Σ we have $|\Sigma|$ possible symbols, so $|\Sigma^5| = |\Sigma|^5 = 3^5 = 243$.

(c) What is the set $\{w_4 w_6 \mid w \in \Sigma^8\}$?

Solution: The fourth and sixth symbols of a length-8 string over Σ can be any element of Σ, so this is the same set as part (a) above: $\{aa, ab, ac, ba, bb, bc, ca, cb, cc\}$.

(d) What is the set $\{w \in \Sigma^* \mid |w| = 0 \text{ or } |w| = 1\}$?

Solution: This is the set of all strings over Σ whose length is either 0 or 1: $\{\varepsilon, a, b, c\}$.

(e) What is the set $\{xx \mid x \in \Sigma\}$?

Solution: This is the set of all strings consisting of an element of Σ repeated twice: $\{aa, bb, cc\}$.

2. (30 pts.) Languages

Let $\Sigma = \{0,1\}$. For each of the following pairs of languages over Σ, state whether they are equal ($=$), one is a proper subset of the other ($\subset$), or if they are incomparable (neither is a subset of the other). If $L_1 \subset L_2$, give an example of a string in L_2 that is not in L_1, or vice versa if $L_2 \subset L_1$; if the languages are incomparable, give an example string from each language that is not in the other.

(a) $L_1 = \Sigma^2 \cup \Sigma^4$
$L_2 = \Sigma^6$

Solution: L_1 is all strings over Σ of length either 2 or 4, while L_2 is all strings over Σ of length 6. These languages are incomparable: for example, aa is in L_1 but not in L_2, and $aaaaaa$ is in L_2 but not in L_1.

(b) $L_1 = \{1\}^*$
$L_2 = \{x \in \Sigma^* \mid |x| \text{ is odd}\}$

Solution: L_1 is all strings containing only 1s, while L_2 is all strings over Σ with odd length. So the languages are incomparable: for example, $11 \in L_1 - L_2$ and $0 \in L_2 - L_1$.

(c) $L_1 = \{x \in \Sigma^8 \mid \exists y \in \Sigma^*.\, x = yyy\}$
$L_2 = \Sigma^3 \cap \Sigma^8$
(N.B. The dot in L_1 simply indicates the end of the quantifier $\exists y \in \Sigma^*$; you can read the condition as "there exists some $y \in \Sigma^*$ such that $x = yyy$".)

Solution: L_1 consists of all length-8 strings over Σ that can be formed by taking some string y and making 3 consecutive copies of it; however, any such string would have to satisfy $8 = |x| = 3|y|$, which is impossible since 8 is not a multiple of 3. So L_1 is the empty set. L_2 consists of all strings over Σ which have length 3 and length 8, which is also impossible, so L_2 is also the empty set. Therefore $L_1 = L_2$.

(d) $L_1 = \Sigma^*$
$L_2 = \cup_{k \geq 1} \Sigma^k$

Solution: L_1 is all strings over Σ, whereas L_2 is all strings over Σ which have length 1 or more. So $L_2 \subset L_1$: the only string in L_1 which is not in L_2 is the empty string ε, which has length 0.

(e) $L_1 = \{x \in \Sigma^* \mid x \text{ is a binary encoding of a prime number}\}$
$L_2 = \{10\} \cup \{x \in \Sigma^* \mid \text{the last symbol of } x \text{ is } 1\}$

Solution: L_1 consists of all binary strings encoding prime numbers. L_2 consists of the string 10, which is the binary encoding of 2, together with all binary strings ending in 1, which are encodings of all the odd numbers. Since all prime numbers except 2 are odd, this means $L_1 \subset L_2$. For example, the string 1001, which is the binary encoding of 9, is in L_2 but not L_1.

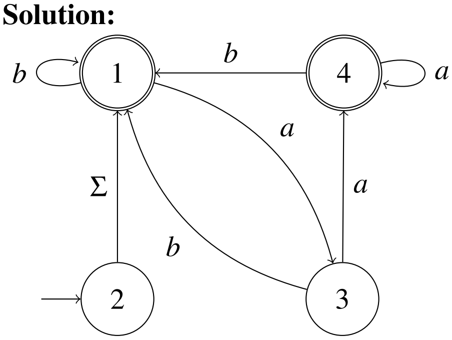
3. (50 pts.) Working with a DFA

Consider the DFA $M = (Q, \Sigma, \delta, q_0, F)$ where:

- $Q = \{1, 2, 3, 4\}$
- $\Sigma = \{a, b\}$
- $\delta(q, s) = \begin{cases} 1 & q = 2 \text{ or } s = b \\ 3 & q = 1 \text{ and } s = a \\ 4 & \text{otherwise} \end{cases}$
- $q_0 = 2$
- $F = \{1, 4\}$

(a) *(20 pts.)* Draw M as a graph.

Solution:



(b) *(18 pts.)* Which of the following strings does M accept: $\varepsilon, b, a, ba, aaa, baab$? Give an accepting path for each accepted string.

Solution:

ε**:** rejected, since the corresponding path 2 ends in a rejecting state (i.e. a non-accepting state);

b**:** accepted, since the corresponding path $2, 1$ ends in an accepting state;

a**:** accepted; the corresponding path $2, 1$ ends in an accepting state;

ba**:** rejected; the corresponding path $2, 1, 3$ ends in a rejecting state;

aaa**:** accepted; the corresponding path $2, 1, 3, 4$ ends in an accepting state;

$baab$**:** accepted; the corresponding path $2, 1, 3, 4, 1$ ends in an accepting state.

(c) *(12 pts.)* What is the language $L(M)$ of M?

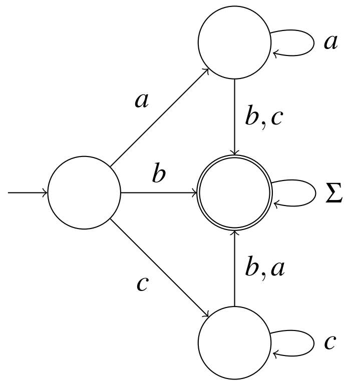
Solution: Note first that from the start state 2, any symbol will take us to 1, and we can never return to 2 since there are no transitions leading to it. From state 1, we can remain there and accept as long as we only see bs, but an a will take us to state 3. State 3 is not accepting, so to accept we either need another b to take us back to state 2, or another a to take us to state 4. In state 4, we will accept as long as we only see as, and we go back to state 1 if we see a b. So after we reach state 1, the only way *not* to accept a string is if we end in state 3, which only happens if we see a single a after a run of bs (if we see at least 2 as, then we'll end up in state 4 and accept). So $L(M)$ consists of all strings which start with either a or b, and then have any sequence of symbols not ending with a single a. As we'll see later, you could write this language as the regular expression $\Sigma\Sigma^*(b|aa)$ (using the fact that any string not ending with a single a either ends in a b or 2 as).

4. (60 pts.) Designing DFAs

For each of the following languages, draw a DFA (as a graph) recognizing it:

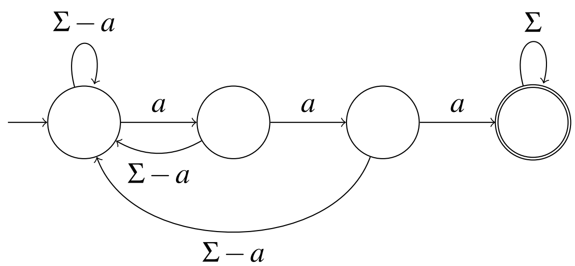
(a) All strings over $\Sigma = \{a, b, c\}$ which either contain a b or contain *both* an a and a c.

Solution:



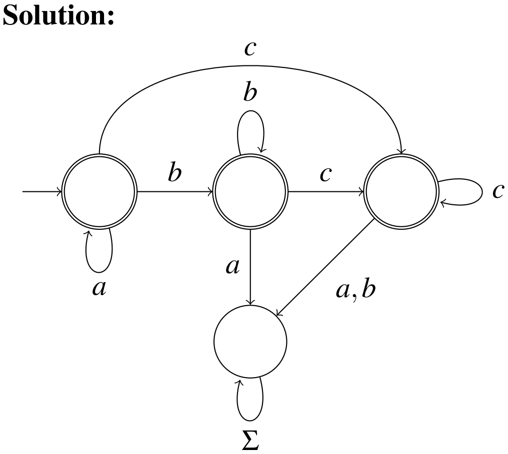
(b) All strings over $\Sigma = \{a, b, c, \ldots, z\}$ containing the substring aaa.

Solution:



(c) All strings over $\Sigma = \{a, b, c\}$ which are in alphabetical order (e.g. *aaac* and *bcc* but not *aba* or *ca*).

Solution:



5. (50 pts.) Counting Strings

We argued in class that given an alphabet Σ with $m = |\Sigma|$ symbols, there are m^n strings of length n: $|\Sigma^n| = m^n$. Let's prove this formally for any $m \geq 1$ and $n \geq 0$ using induction.

(a) Which quantity should we do the induction on?

Solution: There are two obvious choices, m and n. However, n is more natural here because there is a clear relation between the number of strings of length n and those of length $n+1$ for a fixed alphabet size: every string of length $n+1$ is made by appending one more symbol onto the end of a unique string of length n. This recursive relation will allow us to make the inductive case of our argument work, as we'll see below. If we tried to do induction on m, we would need to relate the number of strings over an alphabet of size m to those over an alphabet of size $m+1$ for a fixed length, and this is trickier because the latter aren't built in as simple a way from the former.

If you were able to make a proof by induction on m work, we will of course give you full credit.

(b) Choose a base case and state the claim you need to prove for it.

Solution: Since we want to prove the statement for all $n \geq 0$, our base case should be $n = 0$. Plugging this into the statement we want to prove, we get the claim "Given an alphabet Σ with m symbols, we have $|\Sigma^0| = m^0$" (or $|\Sigma^0| = 1$ if you simplify a bit).

Alternate Solution: If you do induction on n, you could also structure the whole proof as being with respect to a fixed but unspecified alphabet Σ (since you wouldn't be assuming anything about Σ, the proof would then apply for any alphabet). In that case, the claim for the base case would be simply "$|\Sigma^0| = m^0$".

(c) Prove the base case.

Solution: Since Σ^0 is the set of all length-0 strings over Σ, and the only length-0 string is the empty string, we have $|\Sigma^0| = 1 = m^0$ as desired.

(d) State the claim you need to prove for the inductive case, and the inductive hypothesis you can assume while proving it.

Solution: To do (weak) induction, we need to prove that if the statement holds for some $n \geq 0$, then it also holds for $n+1$. Specifically, the claim we need to prove is:

For any alphabet Σ with $m \geq 1$ symbols we have $|\Sigma^{n+1}| = m^{n+1}$.

and the inductive hypothesis we can assume to help prove this claim is:

For any alphabet Π with $p \geq 1$ symbols we have $|\Pi^n| = p^n$.

Notice how we're using two different alphabets in the claim and the inductive hypothesis: since the statement we're trying to prove holds for *all* alphabets, we can assume it for any alphabet Π, not just the specific alphabet Σ that we're trying to prove the claim for.

Alternate Solution: If you structure the proof as working for a fixed but unspecified choice of Σ, then the claim to prove would be the same as above, but the inductive hypothesis would use the same alphabet Σ rather than an arbitrary alphabet Π.

(e) Prove the inductive case.

Solution: Suppose the inductive hypothesis above holds for some $n \geq 0$; then we need to prove the claim above, i.e., for any alphabet Σ with $m \geq 1$ symbols, $|\Sigma^{n+1}| = m^{n+1}$. Observe that for every string in Σ^n there are exactly m strings in Σ^{n+1} obtained by appending one additional symbol from Σ; conversely, every string in Σ^{n+1} is obtained in this way from a unique string in Σ^n. Therefore $|\Sigma^{n+1}| = m|\Sigma^n|$. However, by the inductive hypothesis, we have $|\Sigma^n| = m^n$, so therefore $|\Sigma^{n+1}| = m \cdot m^n = m^{n+1}$ as desired.

(f) Conclude your proof of the original statement.

Solution: By induction, for all $n \geq 0$ and any alphabet Σ with $m \geq 1$ symbols we have $|\Sigma^n| = m^n$.

Alternate Solution: (If you fixed the alphabet Σ in the inductive hypothesis:) By induction, for all $n \geq 0$ we have $|\Sigma^n| = m^n$. Since Σ was arbitrary, the result holds for any alphabet.

N.B. Whenever doing a proof by induction, make sure to include all of the information above. You don't have to break up your presentation into as many pieces as we did here, but you need to clearly state what you're proving by induction, separate the base and inductive cases, etc. For example, I often use language like "We prove this for all $k \in \{0, \ldots, n\}$ by induction on k in decreasing order. In the base case $k = n$, we have...", and then in a new paragraph for the inductive case, "Now suppose the hypothesis holds for $k > 0$; then it also holds for $k - 1$ because...".