18.404/6.5400 Recitation 2

1 Nondeterministic Finite Automata (NFA)

A *nondeterministic finite automaton* is an extension of the deterministic finite automata covered in the first week. The key difference is the nondeterminism, which allows the NFA to exist in multiple states at a single time. If, at the end of a string, any of the states the NFA is on is an accepting state, the NFA will accept the input string. Only if none of the states the NFA is on is an accepting state will the NFA reject. Note that DFAs and NFAs are equivalent in power (a language is recognized by some DFA if and only if the language is recognized by an NFA).

2 More Closure Properties

As defined in class, a language is regular if and only if it can be recognized by some finite automata. Because DFAs are more simple to work with, when proving most closure properties, we usually start with a DFA, and then create an NFA that satisfies the closure property we are trying to prove.

Exercise. Show that regular languages are closed under reversal. Reversal of a language A is defined as

$$A^R = \{w^R | w \in A\}.$$

Proof. Let A be a regular language. Then there exists some DFA M such that $L(M) = A$. We want to construct NFA N that recognizes the reverse of A. At a high-level, this NFA should traverse M backwards, and that requires two steps.

Because we want to accept the reverse of language A, a first step would be to have N's accept state be M's start state, and N's start state be M's accept states. This doesn't work immediately, as we can only have 1 start state, but M could have no or multiple accept states. We fix this by adding an extra state q_0 to be the start state of N, and then adding ϵ-transitions from q_0 to the original accept states of M. Another way to think of this is that since we are looking at strings in reverse order, M could accept a string from any of the accept states, so we want to start at all of them, which we can achieve through ϵ-transitions.

We are not done though! To make sure we can truly work backwards, we need to reverse all the transitions of M by reversing the direction of all the arrows of M. Note that if we had multiple arrows going into a state q in our DFA M, we'll have multiple arrows going out of q in our NFA N. But that's OK through the power of nondeterminism! NFAs allow one state to transition to a set of states, not just one other unique state. For example, if q_1 and q_2 both transition to q_3 when reading a 0 in M, then q_3 will transition to a set of states including q_1 and q_2 when reading a 0. Thus, we've constructed an NFA N, which accepts A^R. □

Exercise. Show that regular languages are closed under intersection.

Proof. Let A and B be regular languages. We want to show that $A \cap B$ must also be regular. To do so, we rely on De Morgan's law, which states that $\overline{A \cup B} = \overline{A} \cap \overline{B}$. We will work backwards to get $A \cap B$ from A and B. By closure under complement, we know that $\overline{A}$ and $\overline{B}$ are regular. Furthermore, $\overline{A} \cup \overline{B}$ and $\overline{\overline{A} \cup \overline{B}}$ are both regular the application of closure

under union then closure under complement. Applying De Morgan's law, $\overline{\overline{A} \cup \overline{B}} = A \cap B$. Therefore $A \cap B$ is regular. $\square$

3 Nonregularity

We've been working with proving that languages are regular - to do this, we need to construct a finite automata that recognizes a given language. Next, we'll practice techniques to show that languages are *nonregular*, meaning that no finite automata will recognize the language. This is where the **pumping lemma** and closure techniques come in handy.

3.1 Pumping Lemma

Formally, the pumping lemma states the following.

Lemma. *If A is a regular language, then there exists a number p (the pumping length) where, for any string $s \in A$ of length at least p, s may be divided into three pieces, $s = xyz$, satisfying the following conditions:*

- *for every whole number i (including 0), $xy^iz \in A$,*
- *$|y| > 0$, and*
- *$|xy| \leq p$.*

The intuition for the pumping lemma is that, because our automata are finite, if a string is accepted by an automata but has more characters than the number of states of the automata, some state must have been repeated at least once. This means that in accepting the string, the automata must have gone through some loop. Repeating this loop multiple times or just deleting the loop entirely would still result in an accepted string, so the loop can be thought of as an analogue for y in the pumping lemma.

The pumping length p can be thought of roughly as the number of states in a DFA that recognizes a given language (this is not exact, and since each language can be accepted by many DFAs all with different number of states, it's difficult to actually give a value of p for a language). Any string with length greater than p has a loop in it, so it can be pumped by repeating or deleting the loop.

In general when using the pumping lemma, we use it as a proof of contradiction. We assume a language is regular, then try to come up with an example string s such that when we repeat or delete y, we wind up with a string not in the language. Below are two proofs of nonregularity using the pumping lemma. The first is explained more in-depth for understanding, and the second is an example of an acceptable write-up on an exam/pset.

Exercise. Show that the language $B = \{0^n1^m0^n \mid n, m \geq 0\}$ is not regular.

Proof. For the sake of contradiction, we assume B is regular. By the pumping lemma, B has a pumping length p. Our high-level strategy is to choose s such that $|s| \geq p$ and when we pump s, we get a string not in B.

Specifically, one key restriction of this language is that the string of zeros at the beginning must be the same length as the string of zeros at the end. So, if we can ensure that y is fully within the string of zeros at the beginning and pump it, our output string will have

the string of zeros at the beginning be longer than the string of zeros at the end, satisfying our contradiction.

To this end, we choose $s = 0^p10^p$. By the conditions of the pumping lemma, and because $|s| = 2p + 1 \geq p$, **for every** splitting $s = xyz$,

$$x = 0^a, y = 0^b, z = 0^c10^p.$$

By the pumping lemma, we also know that $b > 0$. Now, when we pump s by repeating y twice, we get

$$xy^2z = 0^{a+2b+c}10^p = 0^{p+b}10^p \notin B,$$

contradicting B being a regular language. □

Exercise. Show that the language $C = \{0^i1^j \mid i \geq j\}$ is not regular.

Proof. Assume for the sake of contradiction that C is regular, and let p be the pumping length of C. Choose $s = 0^p1^p$. Because $|s| = 2p > p$, apply the pumping lemma to s. Then, for every partition $s = xyz$,

$$x = 0^a, y = 0^b, z = 0^c1^p, b > 0.$$

By repeating y 0 times, we get $s' = xz = 0^{a+c}1^p$. By the pumping lemma, s' should be in C, but $a + c < p$ (as b is greater than 0). Thus, C does not satisfy the pumping lemma, implying that it is not regular. □

3.2 Using Closure Properties for Nonregularity

Some languages are irregular, but at first glance, it seems difficult to use the pumping lemma to prove nonregularity. In these cases, we can assume the language is regular, use one (or more) of the closure properties with a regular language to produce an irregular language, showing by contradiction that the language is irregular.

Exercise. Show that the language $D = \{0^i1^j \mid i \neq j\}$ is irregular.

Proof. Although it is possible to show that this language is irregular using just the pumping lemma, it involves a lot of work, so we will do it using closure properties instead.

Before we begin, we first show that regular languages are closed under difference; if A and B are regular languages, the language consisting of strings that are in A but not in B is a regular language. This is because A set minus B is equivalent to A intersected with the complement of B (in math notation, $A \setminus B = A \cap \bar{B}$). As shown in class, regular languages are closed under both intersection and complement, so regular languages are closed under difference as well.

For the sake of contradiction, assume D is a regular language. We want to show that, using closure properties with a language we can prove is regular, we get a language that is irregular. Let $E = 0^*1^*$, which is a regular language. By closure under difference, $E \setminus D = \{0^i1^i \mid i \geq 0\}$. However, as proved in lecture, $\{0^i1^i \mid i \geq 0\}$ is not regular, contradicting our assumption that D is regular. □

4 Summary

4.1 Closure Properties of Regular Languages

Consider two regular languages A and B. So far, we have shown that all of the following languages must also be regular by closure properties:

- $A \cup B$ (closure under union)
- AB (closure under concatenation)
- A^* (closure under star)
- $\overline{A}$ (closure under complement - proved on pset)
- A^R (closure under reversal)
- $A \cap B$ (closure under intersection)

4.2 Proving Regularity and Nonregularity

These lists are not exhaustive, but they can provide a guideline for you as you approach the problem sets. Given a language A, **we can prove A is regular** by:

1. Constructing a DFA or NFA recognizing A.
2. Providing a regular expression for A (e.g. $A = B \cup C$ where B and C are regular).
3. Proving $\overline{A}$ or A^R is regular.

Note: it's not necessarily true that if $A \cup B$ is regular and B is regular then A is regular. Consider the case when $A = \{0^n 1^n | n \geq 0\}$ and $B = \Sigma^*$. We know B is regular and $A \cup B = \Sigma^*$ is regular, but we have proven that A is not regular. We can apply the closure properties in 3. to prove nonregularity because they are reflective ($\overline{\overline{A}} = A$ and $(A^R)^R = A$).

Given a language A, **we can prove A is not regular** by:

1. Directly applying the pumping lemma.
2. Applying some closure properties to A to get a new language A' (which is also regular), then applying the pumping lemma to A'.