1 Lecture Summary

- We gave a brief overview of inductively defined sets, inductively defined functions, and proofs by structural induction
- We showed how to prove that a machine is correct using induction on strings

Note: I include regular expressions in the examples below, although we did not cover them until the lecture after this one.

2 Inductively defined sets

To define a set inductively, you give a collection of rules for constructing elements in the set from other elements; the set then consists of all elements that can be constructed by applying the rules a finite number of times.

E.g. The set Σ^* of strings is given by the two rules:

1. $\epsilon \in \Sigma^*$
2. If $x \in \Sigma^*$ and $a \in \Sigma$ then $xa \in \Sigma^*$

E.g. The set RE of regular expressions is given by the following rules:

1. $\emptyset \in RE$
2. $\epsilon \in RE$
3. if $a \in \Sigma$ then $a \in RE$
4. if r_1 and r_2 are elements of RE then $r_1 r_2 \in RE$
5. if r_1 and r_2 are elements of RE then $r_1 | r_2 \in RE$
6. if $r \in RE$ then $r^* \in RE$

E.g. The set $\mathbb{N}$ is given by the following rules:

1. $0 \in \mathbb{N}$
2. If $n \in \mathbb{N}$ then $n + 1 \in \mathbb{N}$

3 Inductively defined functions

If X is an inductively defined set, we can define a function f with domain X recursively (I will use the terms "inductive" and "recursive" interchangably) by explaining how to evaluate f for elements constructed with each rule. For the rules that say something like "if $x \in X$ then $xa \in X$" we get to make use of the definition of $f(x)$ in our definition of $f(xa)$.

E.g. we can define the length function $\ell : \Sigma^* \to \mathbb{N}$ by

1. $\ell(\epsilon) = 0$
2. $\ell(xa) = 1 + \ell(x)$ (note we can make use of $\ell(x)$ when defining $\ell(xa)$)

E.g. last lecture we defined $\hat{\delta} : Q \times \Sigma^* \to Q$ inductively by

1. $\hat{\delta}(q, \epsilon) = q$
2. $\hat{\delta}(q, xa) = \delta(\hat{\delta}(q, x), a)$ (note we used $\hat{\delta}(q, x)$)

E.g. we can define the language function $L : RE \to 2^{\Sigma^*}$ by

1. $L(\emptyset) = \emptyset$
2. $L(\epsilon) = \{\epsilon\}$
3. $L(a) = \{a\}$
4. $L(r_1 r_2) = \{xy \mid x \in L(r1) \text{ and } y \in L(r2)\}$ (note the use of $L(r_1)$ and $L(r_2)$)
5. $L(r_1|r_2) = L(r_1) U L(r_2)$ (note use of $L(r_1)$ and $L(r_2)$)
6. $L(r*) = \{x_1 x_2 x_3 \cdots x_n \mid x_i \in L(r)\}$ (note use of $L(r)$)

E.g. we can define the "plus 3" function $p_3 : \mathbb{N} \to \mathbb{N}$ by

1. $p_3(0) = 3$
2. $p_3(n + 1) = p_3(n) + 1$

E.g. we can define the "binary interpretation" function $b : \{0, 1\}^* \to \mathbb{N}$ by

1. b$(\epsilon) = 0$
2. (a) $b(x0) = 2 * b(x)$
   (b) $b(x1) = 2 * b(x) + 1$

(we implicitly use this function below when we say "x represents n"; we really mean $b(x) = n$)

4 Proofs by structural induction

If X is an inductively defined set, we can prove statements of the form "for all $x \in X$, $P(x)$" by giving a proof for each of the rules defining the set X. When giving proofs for the rules that build a new element (e.g. xa) from a pre-existing element (e.g. x), we can assume $P(x)$ in the proof of $P(xa)$.
E.g. Claim: for all strings $x \in \Sigma^*$, $\ell(x) \geq 0$. Proof by induction on the structure of x. We must consider two cases:

1. x is the empty string; we must show $\ell(\epsilon) \geq 0$.

2. x is of the form ya; we must show $\ell(ya) \geq 0$ but we may assume $\ell(y) \geq 0$

Case 1: by definition, $\ell(\epsilon) = 0$, so since $0 \geq 0$, we have $\ell(\epsilon) \geq 0$.

Case 2: Assume that $\ell(y) \geq 0$. We must show $\ell(ya) \geq 0$. By definition, $\ell(ya) = 1 + \ell(y)$, which by our assumption is non-negative.

E.g. Claim: For all regular expressions $r \in RE$, there exists an NFA N such that $L(r) = L(N)$

Proof: see future lectures

5 Proof of correctness of an automata

In the previous lecture, we constructed an automata by associating a fact with each state, and checking that each transition maintained the truth of the fact.

We considered the following problem: to draw an automata that accepts any string that represents a multiple of 3 in binary. We derived the following automata for the language (given here in table form; you are encouraged to check this with your own picture!)

Set of states:

State	Invariant
q_0	x represents a multiple of 3
q_1	x represents $3k + 1$ for some k
q_2	x represents $3k + 2$ for some k

Transition function:

State	Input	New state
q_0	0	q_0
q_0	1	q_1
q_1	0	q_2
q_1	1	q_0
q_2	0	q_1
q_2	1	q_2

We reasoned that the start state should be q_0, because the empty string represents 0, which is a multiple of 3. We alsu argued that the final state should be q_0, because we wish to accept exactly the strings that represent multiples of 3.

How could we prove that this machine does actually accept the correct language?

> Claim: $L(M) = \{x \mid x \text{ represents a multiple of 3 in binary}\}$
> Proof: ?

If we expand the definition of $L(M)$, it becomes clear that induction on the input might be a good approach:

> Claim: for all $x \in \Sigma^*$, $\hat{\delta}(q_0, x) = q_0$ if and only if x represents a multiple of 3 in binary.

To avoid writing that over and over again, let's define $P(x)$ to be that statement:

Let $P(x)$ be the statement "$\hat{\delta}(q_0, x) = q_0$ if and only if x represents a multiple of 3 in binary".

Claim: for all $x \in \Sigma^*$, $P(x)$

Proof: by induction on the structure of x. There are two cases: (1) we must show $P(\epsilon)$, and (2) We must show $P(xa)$, under the assumption $P(x)$.

Case 1: We must show that if $\hat{\delta}(q_0, \epsilon) = q_0$, then ϵ represents a multiple of 3. By definition, ϵ represents 0, so this clearly holds.

Note: this "base case" is exactly the reasoning we went through when deciding where to put the initial state.

Case 2: Assume that (if $\hat{\delta}(q_0, x) = q_0$ then x represents a multiple of 3); we must show that if $\hat{\delta}(q_0, xa) = q_0$ then xa represents a multiple of 3. To show this, let's assume that $\hat{\delta}(q_0, xa) = q_0$. We must now show that xa represents a multiple of 3.

Unfortunately, we don't know enough to finish the proof. The problem is that we can only apply the inductive hypothesis if $\hat{\delta}(q_0, x) = q_0$, but that might not be the case! For example, if x was the string "111" and a was the character '1', then the inductive hypothesis would tell us that if $\hat{\delta}(q_0, x)$ happened to be q_0, then x would be a multiple of 3, but that's not helpful because in fact $\hat{\delta}(q_0, x) = q_1$.

In our informal reasoning for this case, we would have said "since $\hat{\delta}(q_0, x)$ is q_1, we know that x represents $3k + 1$ for some k, so $x1$ must represent $2 \cdot (3k + 1) + 1 = 6k + 3$ which is a multiple of 3".

We'd like to be able to use our inductive hypothesis to say "since $\hat{\delta}(q_0, x)$ is q_1, x must represent $3k+1$ for some k. In fact, we'd like our inductive hypothesis to say that whenever we get to some state, the corresponding invariant holds.

Let's try doing exactly this.

Let $P(x)$ be the following statement: "

(a) if $\hat{\delta}(q_0, x) = q_0$, then x represents $3k$ for some k, AND

(b) if $\hat{\delta}(q_0, x) = q_1$, then x represents $3k + 1$ for some k, AND

(c) if $\hat{\delta}(q_0, x) = q_2$, then x represents $3k + 2$ for some k."

Claim: For all $x \in \Sigma^*$, $P(x)$

Proof: by induction on the structure of x. We must consider two cases:

Case 1: We must show $P(\epsilon)$. By definition of $\hat{\delta}$, $\hat{\delta}(q_0, \epsilon) = q_0$, so parts (b) and (c) are vacuously true. That is, they both say "if (false) then (something)", which is always true. Part (a) is true because ϵ represents $3 \cdot 0$.

Case 2: We must show $P(xa)$, assuming the inductive hypothesis $P(x)$. We must show all three parts of $P(xa)$.

(a) Suppose that $\hat{\delta}(q_0, xa) = q_0$. We want to show that xa represents $3k$ for some k. By looking at the definition of $\hat{\delta}$ and δ, we see that there are two possible ways for $\hat{\delta}(q_0, xa) = q_0$; either

i. $\hat{\delta}(q_0, x) = q_0$ and $a = 0$. In this case, our inductive hypothesis says that since $\hat{\delta}(q_0, x) = q_0$, that x represents $3k$ for some k. Therefore, xa represents $2 \cdot 3k = 6k = 3 \cdot (2k)$. Therefore xa represents $3k$ for some k. Or,

ii. $\hat{\delta}(q_0, x) = q_1$ and $a = 1$. In this case, our inductive hypothesis says that since $\hat{\delta}(q_0, x) = q_1$, that x represents $3k + 1$ for some k. Therefore, xa represents $2 \cdot (3k + 1) + 1 = 6k + 3 = 3 \cdot (k + 1)$. Therefore xa represents $3k$ for some k.

In both cases we have shown that xa represents $3k$ for some k, which is what was required for part (a).

(b) Suppose that $\hat{\delta}(q_0, xa) = q_1$. We want to show that xa represents $3k + 1$ for some k. Again the definition of $\hat{\delta}$ and δ shows us that there are two ways for this to happen, namely,

i. $\hat{\delta}(q_0, x) = q_0$ and $a = 1$. In this case, the inductive hypothesis says [**completion of this case left as exercise**] and therefore xa represents $3k + 1$ for some k. Or,

ii. $\hat{\delta}(q_0, x) = q_2$ and $a = 0$. [**left as exercise**]

In both cases we have shown that xa represents $3k + 1$ for some k, which is what was required for part (b).

(c) [**this part is left as an exercise**]

Since we have shown parts (a), (b), and (c) of $P(xa)$ hold, we conclude that $P(xa)$ holds.

Since we have shown $P(\epsilon)$ and $P(xa)$, we conclude that for any string x, $P(x)$ holds.

And the proof works out! I've left some of the details to you; I encourage you to work them out to make sure you see what's going on. Also note that this proof is just a careful encoding of the informal reaonsing we did as we were designing the machine in the first place. You'll find that the complete proof has 6 sub-cases in the inductive step; one for each of the 6 transitions in the machine, and that the proofs of each of these subcases is exactly like the reasoning we did while drawing the corresponding transition.

Although this proof has lots of cases and details, it follows a very simple structure. The key ideas are:

1. for each state q, write down a fact that you know about x if $\hat{\delta}(q_0, x) = q$.

2. Combine all of these facts into $P(x)$

3. Do an inductive proof of $P(x)$. You'll find that this will force you to check that every possible transition is correct.