Finite Automata Theory and Formal Languages

TMV027/DIT321– LP4 2013

Lecture 3
Ana Bove

March 21st 2013

Overview of today's lecture:

- Formal Proofs;
- Inductively defined sets;
- Proofs by (structural) induction.

How Formal a Proof Should Be?

- Should be convincing;
- Should not leave too much out;
- The validity of each step should be easily understood.

Valid steps are for example:

- Reduction to definition:

  "x is a positive integer" is equivalent to "$x > 0$";
- Use of hypothesis;
- Combining previous facts and known statements:

  "Given $A \Rightarrow B$ and A we can conclude B by *modus ponens*".

Form of Statements

Statements we want to prove are usually of the form

$$\text{If } H_1 \text{ and } H_2 \ldots \text{and } H_n \text{ then } C_1 \text{ and } \ldots \text{and } C_m$$

or

$$P_1 \text{ and } \ldots \text{and } P_k \text{ iff } Q_1 \text{ and } \ldots \text{and } Q_m$$

for $n \geqslant 0$; $m, k \geqslant 1$.

Note: Observe that one proves the *conclusion* assuming the validity of the *hypotheses*!

Different Kinds of Proofs

Proofs by Contradiction

$$\text{If } H \text{ then } C$$

is logically equivalent to

$$H \text{ and not } C \text{ implies "something known to be false".}$$

Example: If $x \neq 0$ then $x^2 \neq 0$.

Proofs by Contrapositive

"If H then C" is logically equivalent to "If not C then not H"

Proofs by Counterexample

We find an example that "breaks" what we want to prove.

Example: All natural numbers are odd.

Proofs by Induction

How to prove an statement over the natural numbers?

Mathematical induction: When we want to prove a property P over *all* natural numbers.
Given $P(0)$ and $\forall n \in \mathbb{N}.\ P(n) \Rightarrow P(n+1)$ then $\forall n \in \mathbb{N}.\ P(n)$.

More generally: given $P(i), P(i+1), \ldots, P(j)$ for $j > i$, and $\forall n \geqslant i.\ P(n) \Rightarrow P(n+1)$ then $\forall n \geqslant i.\ P(n)$.

Course of value induction: Variant of mathematical induction.
Given $P(i), P(i+1), \ldots, P(j)$ for $j > i$ and $\forall i \leqslant m < n.\ P(m) \Rightarrow P(n)$ then $\forall n \geqslant i.\ P(n)$.

Hypotheses in read are called our *inductive hypotheses* (IH).

Example: Proof by Induction

Proposition: *Let $f(0) = 0$ and $f(n+1) = f(n) + n + 1$. Then, $\forall n \in \mathbb{N}$, we have $f(n) = n(n+1)/2$.*

Proof: By induction on n where $P(n)$ is $f(n) = n(n+1)/2$.

Base case: We prove that $P(0)$ holds.

Inductive step: We prove that if for a given $n \geqslant 0$ $P(n)$ holds (our IH), then $P(n+1)$ also holds.

Closure: Now we have established that for *all* n, $P(n)$ is true!
In particular we know that $P(0), P(1), P(2), \ldots, P(15), \ldots$ hold.

Example: Proof by Induction

Proposition: *If $n \geqslant 8$ then n can be written as a sum of 3's and 5's.*

Proof: By course of value induction on n where $P(n)$ is "n can be written as a sum of 3's and 5's".

Base cases: $P(8), P(9)$ and $P(10)$ hold.

Inductive step: Now we want to prove that if $P(8), P(9), \ldots, P(n)$ hold for $n \geqslant 10$ (our IH) then $P(n+1)$ holds.

Observe that if $n \geqslant 10$ then $n \geqslant n+1-3 \geqslant 8$.
Hence by inductive hypothesis $P(n+1-3)$ holds.
By adding an extra 3 then $P(n+1)$ holds as well.

Closure: $\forall n \geqslant 8.\ P(n)$.

Example: Proof by Induction

Proposition: *All horses have the same colour.*

Proof: Let $P(n)$ be "in any set of n horses they all have the same colour".

Base cases: $P(0)$ is not interesting in this example.
$P(1)$ is clearly true.

Inductive step: Let us show that $P(n)$ (our IH) implies $P(n+1)$.
Let $h_1, h_2, \ldots, h_n, h_{n+1}$ be a set of $n+1$ horses.
Take $h_1, h_2, \ldots, h_n$. By IH they all have the same colour.
Take now $h_2, h_3, \ldots, h_n, h_{n+1}$. Again, by IH they all have the same colour.
Hence, by transitivity, all horses $h_1, h_2, \ldots, h_n, h_{n+1}$ must have the same colour.

Closure: $\forall n.\ P(n)$ has n horses with the same colour.

What went wrong???

Mutual Induction

Sometimes we cannot prove a single statement $P(n)$ but rather a group of statements $P_1(n), P_2(n), \ldots, P_k(n)$ simultaneously by induction on n.

This is very common in automata theory where we need an statement for each of the states of the automata.

Example: Recall the on/off-switch from last lecture.

Example: Proof by Mutual Induction

Let $f, g, h : \mathbb{N} \to \{0, 1\}$ be as follows:

$$\begin{array}{lll} f(0) = 0 & g(0) = 1 & h(0) = 0 \\ f(n+1) = g(n) & g(n+1) = f(n) & h(n+1) = 1 - h(n) \end{array}$$

Proposition: $\forall n.\ h(n) = f(n)$.

Proof: If $P(n)$ is "$h(n) = f(n)$" it does not seem possible to prove $P(n) \Rightarrow P(n+1)$ directly.

We strengthen $P(n)$ to $P'(n)$ as follows:

Let $P'(n)$ be "$h(n) = f(n) \wedge h(n) = 1 - g(n)$".

We prove $P'(0)$, that is, $h(0) = f(0) \wedge h(0) = 1 - g(0)$.

Then we prove that $P'(n+1)$ follows from $P'(n)$ (our IH).

Now we know that $\forall n.\ P'(n)$ is true and hence $\forall n.\ P(n)$ is true.

Application to Automata

We can think of f, g and h as a *circuit*.

The circuit can be represented as an automaton as follows:

- The states are the possible values of $s(n) = (f(n), g(n), h(n))$;
- The transitions are from the states $s(n)$ to the state $s(n+1)$;
- Initial state is $s(0) = (0, 1, 0)$.

One can check the invariant $f(n) = h(n)$ on all the states accessible from the initial state.

Inductively Defined Sets

Natural Numbers: Base case: 0 is a natural number;
Inductive step: If n is a natural number then $n + 1$ is a natural number;
Closure: There is no other way to construct natural numbers.

Finite Lists: Base case: [] is the empty list over any set A;
Inductive step: If $a \in A$ and xs is a list over A then $a : xs$ is a list over A;
Closure: There is no other way to construct lists.

Finitely Branching Trees: Base case: () is a tree over any set A;
Inductive step: If $t_1, \ldots, t_k$ are tree over the set A and $a \in A$, then $(a, t_1, \ldots, t_k)$ is a tree over A;
Closure: There is no other way to construct trees.

$\vdots$

Compare this with the definition of (recursive) data types in a programming language! *What can you say?*

Inductively Defined Sets (Cont.)

To define a set S by induction we need to specify:

Base cases: Here we say which specific elements $e_1, \dots, e_m$ belong to S.

Inductive steps: Assuming that $s_1, \dots, s_n$ belong to S, we indicate how to use $s_1, \dots, s_n$ in order to construct new elements of S $\mathsf{c}_1[s_1, \dots, s_n], \dots, \mathsf{c}_\mathsf{k}[s_1, \dots, s_n]$.

Closure: There is no other way to construct elements in S.

Example: The set of simple Boolean expressions is defined as:

Base cases: true and false are Boolean expressions
Inductive steps: if a and b are Boolean expressions then

$$(a) \qquad \text{not } a \qquad a \text{ and } b \qquad a \text{ or } b$$

are also Boolean expressions.
Closure: ... (We will usually omit this part.)

Proofs by Structural Induction

Generalisation of mathematical induction to other inductively defined object such as lists, trees, ...

VERY useful in computer science since it allows to prove properties over the (finite) elements in a data type!

Given an inductively defined set S, to prove $\forall s \in S.\ P(s)$ then:

Base cases: We prove that P holds for all base cases: $P(e_1), \dots, P(e_m)$.

Inductive steps: Assuming that $P(s_1), \dots, P(s_n)$ hold (our *inductive hypotheses* IH), we prove that $P(\mathsf{c}_1[s_1, \dots, s_n]), \dots, P(\mathsf{c}_\mathsf{k}[s_1, \dots, s_n])$ also hold.

Closure: $\forall s \in S.\ P(s)$.

Example: Proof by Structural Induction

We can now use recursion to define functions over an inductively defined set and then prove properties of these functions by structural induction.

Given the finite lists, let us (recursively) define the append and length functions:

$$[] +\!+\ ys = ys \qquad\qquad \text{len } [] = 0$$
$$(a : xs) +\!+\ ys = a : (xs +\!+\ ys) \qquad\qquad \text{len } (a : xs) = 1 + \text{len } xs$$

Proposition: $\forall xs, ys.\ \text{len } (xs +\!+\ ys) = \text{len } xs + \text{len } ys.$

Proof: By structural induction on xs.
Base case: We prove $P[]$.
Inductive step: We show that $P(xs)$ implies $P(a : xs)$.
Closure: $\forall xs.\ P(xs)$.

Example: Proof by Structural Induction

Given the finitely branching trees, let us (recursively) define functions counting the number of edges and of nodes:

$$\text{ne}() = 0 \qquad\qquad \text{nn}() = 1$$
$$\text{ne}(a, t_1, \ldots, t_k) = k + \text{ne}(t_1) + \ldots + \text{ne}(t_k) \qquad\qquad \text{nn}(a, t_1, \ldots, t_k) = 1 + \text{nn}(t_1) + \ldots + \text{nn}(t_k)$$

Proposition: $\forall t.\ \text{nn}(t) = 1 + \text{ne}(t).$

Proof: By structural induction on t.
Base case: We prove $P()$.
Inductive step: We show that if $P(t_1), \ldots, P(t_k)$ then $P(a, t_1, \ldots, t_k)$.
Closure: $\forall t.\ P(t)$.

Proofs by Induction: Steps to Follow

1. State property P to prove.
   Might be more general than the actual statement we need to prove.
2. **Determine and state the method to use in the proof!!!!**
   Example: (Mathematical) Induction on the length of the list, course by value induction on the height of a tree, structural induction on the structure of certain data type, ...
3. Identify and state base case(s).
   Could be more than one! Not always trivial to determine.
4. Prove base case(s).
5. Identify and state IH!
   Will depend on the method to be used (see point 2).
6. Prove inductive step.
7. (State closure.)
8. Deduce your statement from P (if not the same).

Overview of Next Lecture

Sections Sections 2–2.2.

- Deterministic Finite Automata (DFA).