Finite Automata Theory and Formal Languages

TMV027/DIT321– LP4 2014

Lecture 3
Ana Bove

March 20th 2014

Overview of today's lecture:

- Formal proofs;
- Mathematical and course-of-value induction;
- 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*!

Example: We can easily prove "if $0 = 1$ then $4 = 2.000$".

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.

Proving a Property over the Natural Numbers

How to prove an statement over *all* the Natural numbers?

Example: $\forall n \in \mathbb{N}$. if $n \mid 4$ then $n \mid 2$.

First we need to look at what the Natural numbers are ...

They are an *inductively defined set* and can be defined by the following *two* rules:

$$\frac{}{0 \in \mathbb{N}} \qquad \frac{n \in \mathbb{N}}{n+1 \in \mathbb{N}}$$

(More on inductively defined sets on page 16.)

Mathematical Induction

Given $P(0)$ *and* $\forall n \in \mathbb{N}.\ P(n) \Rightarrow P(n+1)$ *then* $\forall n \in \mathbb{N}.\ P(n)$.

$$\frac{\overbrace{P(0)}^{\text{base case}} \qquad \overbrace{\forall n \in \mathbb{N}.\ P(n) \Rightarrow P(n+1)}^{\text{inductive step}}}{\underbrace{\forall n \in \mathbb{N}.\ P(n)}_{\text{statement to prove}}}$$

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 $\forall n \geqslant i.\ P(n)$.

$$\frac{P(i), P(i+1), \ldots, P(j) \qquad \forall n \geqslant i.\ P(n) \Rightarrow P(n+1)}{\forall n \geqslant i.\ P(n)}$$

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

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 $\forall n \geqslant i.\ P(n)$.

$$\frac{P(i), P(i+1), \ldots, P(j) \qquad \forall i \leqslant m < n.\ P(m) \Rightarrow P(n)}{\forall n \geqslant i.\ P(n)}$$

Example: Proof by Induction

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

Proof: By *mathematical induction* on n.

Let $P(n)$ be $f(n) = n(n+1)/2$.

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, $P(0), P(1), P(2), \ldots, P(15), \ldots$ 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.

Let $P(n)$ be"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: We shall 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: $\forall n \geqslant 8$. n can be written as a sum of 3's and 5's.

Example: All Horses have the Same Colour

Example: Proof by Induction

Proposition: *All horses have the same colour.*

Proof: By *mathematical induction* on n.

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: $\forall n$. all the n horses in the set have the same colour.

Example: 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 first 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 $P'(0) : h(0) = f(0) \wedge h(0) = 1 - g(0)$.

Then we prove that $P'(n) \Rightarrow P'(n+1)$.

Since $\forall n.\ P'(n)$ is true then $\forall n.\ P(n)$ is true.

Application to Automata

We can think of f, g and h as the functional representation of an automaton of the following *circuit*:

- 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.

(All this will make more sense once we know more about automata.)

Recursive Data Types

How do you define Natural numbers, lists, trees, ... in your favourite programming language?

This is how you would defined them in Haskell:

```
data Nat = Zero | Succ Nat

data List a = Nil | Cons a (List a)

data BTree a = Leaf a | Node a (BTree a) (BTree a)
```

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: (a) 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.

$\vdots$

Inductively Defined Sets (Cont.)

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

Base cases: $e_1, \ldots, e_m \in S$;

Inductive steps: Given $s_1, \ldots, s_n \in S$,
then $c_1[s_1, \ldots, s_n], \ldots, c_k[s_1, \ldots, s_n] \in S$;

Closure: There is no other way to construct elements in S.
(We will usually omit this part.)

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.

Proofs by Structural Induction

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

VERY useful in computer science: 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 $P(e_1), \ldots, P(e_m)$;

Inductive steps: Assuming $P(s_1), \ldots, P(s_n)$ (our *inductive hypotheses* IH), we prove $P(c_1[s_1, \ldots, s_n]), \ldots, P(c_k[s_1, \ldots, s_n])$;

Closure: $\forall s \in S.\ P(s)$.
(We will usually omit this part.)

Inductive Sets and Structural Induction

Inductive definition of S:

$$\frac{}{e_1 \in S} \quad \ldots \quad \frac{}{e_m \in S} \qquad \frac{s_{1_1}, \ldots, s_{1_n} \in S}{c_1[s_{1_1}, \ldots, s_{1_n}] \in S} \quad \ldots \quad \frac{s_{k_1}, \ldots, s_{k_n} \in S}{c_k[s_{k_1}, \ldots, s_{k_n}] \in S}$$

Inductive principle associated to S:

$$\text{base cases} \begin{cases} P(e_1) \\ \vdots \\ P(e_m) \end{cases}$$

$$\text{inductive steps} \begin{cases} \forall s_{1_1}, \ldots, s_{1_n} \in S.\ P(s_{1_1}) \cdots P(s_{1_n}) \Rightarrow P(c_1[s_{1_1}, \ldots, s_{1_n}]) \\ \vdots \\ \forall s_{k_1}, \ldots, s_{k_n} \in S.\ P(s_{k_1}) \cdots P(s_{k_n}) \Rightarrow P(c_k[s_{k_1}, \ldots, s_{k_n}]) \end{cases}$$

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

Example: Structural Induction over Lists

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

Let us (recursively) define the append and length functions over lists:

$$\begin{array}{ll} [] +\!\!+\ ys = ys & \text{len}\ [] = 0 \\ (a : xs) +\!\!+\ ys = a : (xs +\!\!+\ ys) & \text{len}\ (a : xs) = 1 + \text{len}\ xs \end{array}$$

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

Proof: By structural induction on xs.
$P(xs)$ is $\forall ys \in \text{List}\ A.\text{len}\ (xs +\!\!+\ ys) = \text{len}\ xs + \text{len}\ ys$.
Base case: We prove $P[]$.
Inductive step: We show that for all xs, $P(xs)$ implies $P(a : xs)$.
Closure: $\forall xs \in \text{List}\ A.\ P(xs)$.

Example: Structural Induction over Trees

Let us (recursively) define functions counting the number of edges and of nodes of a tree:

$$\begin{array}{ll} \text{ne}(a) = 0 & \text{nn}(a) = 1 \\ \text{ne}(a, t_1, \ldots, t_k) = k + & \text{nn}(a, t_1, \ldots, t_k) = 1 + \\ \quad \text{ne}(t_1) + \ldots + \text{ne}(t_k) & \quad \text{nn}(t_1) + \ldots + \text{nn}(t_k) \end{array}$$

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

Proof: By structural induction on t.
$P(t)$ is $\text{nn}(t) = 1 + \text{ne}(t)$.
Base case: We prove $P(a)$.
Inductive step: We show that for all $t_1, \ldots, t_k$, if $P(t_1), \ldots, P(t_k)$ then $P(a, t_1, \ldots, t_k)$.
Closure: $\forall t \in \text{Tree}\ A.\ P(t)$.

Proofs by Induction: Overview of the 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-of-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(s).
7. (State closure.)
8. Deduce your statement from P (if not the same).

Overview of Next Lecture

Sections 2–2.2:

- DFA: deterministic finite automata.