Finite Automata Theory and Formal Languages
Föreläsning 3 - Proofs

Erik Sjöström

March 24, 2016

1 How Formal Should a Proof Be?

Depends on the purpose but:

- 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 hypotheses
- Combining previous facts in a valid way:
  - "Given $A \Rightarrow B$ and A we can conclude B by *modus ponens*"

2 Form of Statements

Statements we want to prove are usually of the form:

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

or:

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

for $n \geq 0$; $m, k \geq 1$

Anmärkning.
Observe that one proves the conclusion *assuming the validity of the* hypotheses*!*

Exempel 2.1.
We can easily prove "if 0 = 1 then 4 = 2.000"

3 Different Kinds of Proofs

Proofs by contradiction:

If H then C

is logically equivalent to

H and not C implies "something known to be false"

Exempel 3.1.
If $x \neq 0$ *then* $x^2 \neq 0$*, vs ,*$x \neq 0$ *and* $x^2 = 0$ *is impossible!*

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.

Exempel 3.2.
All Natural numbers are odd.

4 Proving a Property over the Natural Numbers

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

Exempel 4.1.
$\forall n \in \mathbb{N}$*, if* $n \mid 4$ *then* $n \mid 2$.

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

Definition 4.1. *They are an* inductively defined set *and can be defined by the following* two *rules:*

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

5 Mathematical/Simple Induction

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

More generally:

$$\frac{P(i), P(i+1), ..., P(j) \qquad \forall i \leq j \leq n,\, \color{red}{P(n)} \Rightarrow P(n+1)}{\forall i \leq n,\, P(n)}$$

Hypotheses in red is called *inductive hypotheses.* (IH).

6 Course-of-Values/Strong Induction

Variant of mathematical induction.

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

Or more generally:

$$\frac{P(i), P(i+1), ..., P(j) \qquad \forall j < n,\, (\forall m,\, i \leq m \leq n \Rightarrow \color{red}{P(m)}) \Rightarrow P(n)}{\forall i \leq n,\, P(n)}$$

Here we might have several inductive hypotheses P(m)!

7 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 $P(n)$ b $f(n) = n(n+1)/2$

- **Base case:** We prove that $P(0)$ holds
- **Inductive step:** We prove that if for a given $n \geq 0$ $P(n)$ holds (our IH) then $P(n+1)$ also holds
- **Closure:** Now we have established that for all n, $P(n)$ is true!

□

8 Example: Proof by Induction

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

Proof. By course-of-values induction on n
Let $P(n)$ be "n can be written as a sum of 3's and 5's"

- **Base case:** $P(8)$, $P(9)$ and $P(10)$ hold
- **Inductive step:**
  - We shall prove that if $P(8)$, $P(9)$, $P(10)$,..., $P(n)$ hold for $n \geq 10$ (our IH) then $P(n+1)$ holds
  - Observe that if $n \geq 10$ then $n \geq n + 1 - 3 \geq 8$
  - Hence by inductive hypotheses $P(n + 1 - 3)$ holds
  - By adding an extra 3 then $P(n+1)$ holds as well
- **Closure:** $\forall n \geq 8$, n can be written as a sum of 3's and 5's.

□

9 Example: All Horses have the Same Colour

Proposition: All horses have the same colour.

Proof. By mathemathical induction on n.
Let $P(n)$ be *"in any set of n horses they all have the same colour"*

- **Base case:**
  - $P(0)$ 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 $h_1, h_2, ..., h_n, h_{n+1}$ be a set of $n+1$ horses
  - Take $h_1, h_2, ..., h_n$. By IH they all have the same colour.
  - Now take $h_2, h_3, ..., h_n, h_{n+1}$. Again by IH they all have the same colour.
  - Hence, by transitivity all horses $h_1, h_2, ..., h_n, h_{n+1}$, must have the same colour
- **Closure:** $\forall n$, all n horses in the set have the same colour.

□

10 Mutual induction

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

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

11 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 no seem possible to prove $P(n) \Rightarrow P(n+1)$ directly.

- We strengthen *P(n)* to *P'(n)*: Let $P'(n)$ be $h(n) = f(n) \wedge h(n) = 1 - g(n)$
- By mathematical induction 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.

□

12 Recursive Data Types

What are (the data types of) Natural numbers, lists, trees, ...?

This is how you would defined them in Haskell:

```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)
```

13 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 a Natural number

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:** If $a \in A$ then *(a)* is a tree over any set A
- **Inductive step:** If $t_1, ..., t_k$ are trees over ther set A and $a \in A$ then $(a, t_1, ..., t_k)$ is a tree over A

- **Closure:** There is no other way to construct trees.

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

- **Base case:** $e_1, ..., e_m \in S$
- **Inductive steps:** Five $s_1, ..., s_n \in S$ then $c_1[s_1, ..., s_{n_1}], ..., c_k[s_1, ..., s_{n_k}]$
- **Closure:** There is no other way to construct elements in S. (We will usually omit this part.)

Exempel 13.1.
The set of simple Boolean expressions if defined as:

- ***Base cases:*** true *and* false *are Boolean expressions*
- *if* a *and* b *are Boolean expressions then:*

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

are also Boolean expressions

14 Proofs by Structural Induction

Generalisation of mathematical induction to other inductively defined sets such as lists, trees, ...
Very useful in computer science: it allows to prove properties over the (finite) elments in a data type!

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

- **Base cases:** We prove that $P(e_1), ..., P(e_m)$
- **Inductive steps:** Assuming $P(s_1), ..., P(s_m)$ (out inductive hypotheses IH), we prove $P(c_1[s_1, ..., s_{n_1}], ..., P(c_k[s_1, ..., s_{n_k}])$
- **Closure:** $\forall s \in S$, $P(s)$ (We will usually omit this part)

15 Inductive Sets and Structural Induction

Inductive definition of S:

$$\frac{}{e_1 \in S} \cdots \frac{}{e_m \in S} \qquad \frac{s_1, ..., s_{n_1} \in S}{c_1[s_1, ..., s_{n_1}] \in S} \cdots \frac{s_1, ..., s_{n_k} \in S}{c_k[s_1, ..., s_{n_k}] \in S}$$

Inductive principle associated to S:

$$\frac{\text{base cases: } \begin{cases} P(e_1) \\ \vdots \\ P(e_m) \end{cases} \qquad \text{inductive steps: } \begin{cases} \forall s_1, ..., s_{n_1} \in S,\ P(s_1), ..., P(s_{n_1}) \Rightarrow P(c_1[s_1, ..., s_{n_1}]) \\ \vdots \\ \forall s_1, ..., s_{n_k} \in S,\ P(s_1), ..., P(s_{n_k}) \Rightarrow P(c_k[s_{k_1}, ..., s_{n_k}]) \end{cases}}{\forall s \in S,\ P(s)}$$

16 Example: Structural Induction over Lists

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

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

$$[] + +ys = ys \qquad len[] = 0$$
$$(a : xs) + +ys = a : (xs + +ys) \qquad len(a : xs) = 1 + lenxs$$

```
[] ++ ys = ys                    len [] = 0
(a:xs) ++ ys = a:(xs ++ ys)   len(a:xs) = 1 + len xs
```

Proposition: $\forall xs, ys \in$ List A, len $(xs + +ys) =$ len $xs +$ len ys

Proof. By structural induction on $xsin$ List A
$P(xs)$ is $\forall ys \in$ List A, len $(xs + +ys) =$ len $xs +$ len ys

- **Base case:** We prove $P[]$
- **Inductive step:** We show $\forall xs \in$ A , $a \in A$, $P(xs) \Rightarrow P(a : xs)$
- **Closure:** $\forall xs \in$ List A, $P(xs)$

□

17 Example: Structural Induction over Lists

Let us (recursively) define append and reverse function over lists:

```
[] ++ ys = ys                    rev [] = []
(a:xs) ++ ys = a:(xs ++ ys)   rev(a:xs) = rev xs ++ [a]
```

Assume append is associative and that ys ++ [] = ys
Proposition: $\forall xs, ys \in$ List A, rev $(xs + +ys) =$ rev $ys + +xs$

Proof. By structural induction on $xs \in$ List A $P(xs)$ is $\forall ys \in$ List A, rev $(xs + +ys) =$ rev $ys + +$ rev xs

- **Base case:** We prive $P[]$
- **Inductive step:** We show $\forall xs \in$ List A, $a \in A$, $P(xs) \Rightarrow P(a : xs)$
- **Closure:** $\forall xs \in$ List A, $P(xs)$

□

18 Example: Structural Induction over Trees

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

```
ne(a) = 0 nn(a) = 1                  nn(a) = 1
ne(a, t_1, ..., t_k) = k +           nn(a, t_1, ..., t_k) = 1 +
          ne(t_1) + ... + ne(t_k)              nn(t_1) + ... + nn(t_k)
```

Proposition: $\forall t \in$ Tree A, *nn(t) = 1 + ne(t)*

Proof. By structural induction on $t \in$ Tree A
$P(t)$ is *nn(t) = 1 + ne(t)*

- **Base case:** We prove *P(a)*
- **Inductive step:** We show $\forall t_1, ..., t_k \in$ Tree A, $a \in A$, $P(t_1), ..., p(t_k) \Rightarrow P(a, t_1, ..., t_k)$
- **Closure:** $\forall t \in$ Tree A, *P(t)*

□

19 Proofs by Induction: Overview of the Steps to Follow

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