Script generated by TTT

Title: FDS (24.05.2019)

Date: Fri May 24 08:30:34 CEST 2019

Duration: 98:18 min

Pages: 56

∀ and ∃ introduction

show $\forall x.\ P(x)$
proof
  fix x local fixed variable
  show $P(x)$ $\langle proof \rangle$
qed

show $\exists x.\ P(x)$
proof
  ⋮
  show $P(witness)$ $\langle proof \rangle$
qed

∃ elimination: **obtain**

$\exists$ elimination: **obtain**

have $\exists\, x.\ P(x)$
then obtain x **where** *p:* $P(x)$ **by** *blast*
$\vdots$ $\quad x$ fixed local variable

$\exists$ elimination: **obtain**

have $\exists\, x.\ P(x)$
then obtain x **where** *p:* $P(x)$ **by** *blast*
$\vdots$ $\quad x$ fixed local variable

Works for one or more x

obtain example

lemma $\neg\ surj(f :: {'a} \Rightarrow {'a}\ set)$
proof
  assume $surj\ f$
  hence $\exists\, a.\ \{x.\ x \notin f\ x\} = f\ a$ **by**$(auto\ simp{:}\ surj\_def)$

obtain example

lemma $\neg\ surj(f :: {'a} \Rightarrow {'a}\ set)$
proof
  assume $surj\ f$
  hence $\exists\, a.\ \{x.\ x \notin f\ x\} = f\ a$ **by**$(auto\ simp{:}\ surj\_def)$
  then obtain a **where** $\{x.\ x \notin f\ x\} = f\ a$ **by** *blast*

obtain example

lemma $\neg$ *surj*$(f :: 'a \Rightarrow 'a\ set)$
proof
  assume *surj f*
  hence $\exists\, a.\ \{x.\ x \notin f\ x\} = f\ a$ **by**$(auto\ simp\text{: } surj\_def)$
  then obtain a **where** $\{x.\ x \notin f\ x\} = f\ a$ **by** *blast*

obtain example

lemma $\neg$ *surj*$(f :: 'a \Rightarrow 'a\ set)$
proof
  assume *surj f*
  hence $\exists\, a.\ \{x.\ x \notin f\ x\} = f\ a$ **by**$(auto\ simp\text{: } surj\_def)$
  then obtain a **where** $\{x.\ x \notin f\ x\} = f\ a$ **by** *blast*
  hence $a \notin f\ a \longleftrightarrow a \in f\ a$ **by** *blast*

Set equality and subset

show $A = B$
proof
  show $A \subseteq B$ $\langle proof \rangle$
next
  show $B \subseteq A$ $\langle proof \rangle$
qed

Set equality and subset

show $A = B$
proof
  show $A \subseteq B$ $\langle proof \rangle$
next
  show $B \subseteq A$ $\langle proof \rangle$
qed

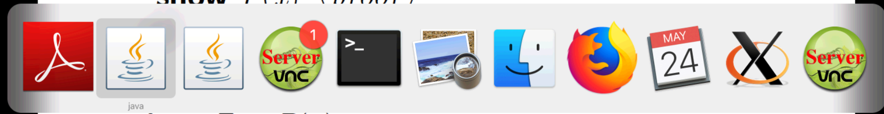
show $A \subseteq B$
proof
  fix x
  assume $x \in A$
  $\vdots$
  show $x \in B$ $\langle proof \rangle$
qed

Isar_Demo.thy

Exercise

∀ and ∃ introduction

show $\forall x.\ P(x)$
proof
  fix x local fixed variable
  show $P(x)$ $\langle proof \rangle$



show $\exists x.\ P(x)$
proof
  ⋮
  show $P(witness)$ $\langle proof \rangle$
qed

Isar_Demo.thy (~/Teaching/FDS/SS19/fds_ss19/Demos/)

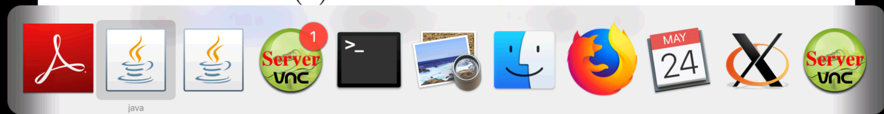
```
qed


text{* Interactive exercise: *}

lemma assumes "∃x. ∀y. P x y" shows "∀y. ∃x. P x y"
proof

qed
sorry


subsection ‹(In)Equation Chains›

lemma "(0::real) ≤ x^2 + y^2 - 2*x*y"
proof -
  have "0 ≤ (x - y)^2" by simp
```

Output Query Sledgehammer Symbols

139,52 (2340/5680)   (isabelle,isabelle,UT

∃ elimination: **obtain**

have $\exists x.\ P(x)$



∀ and ∃ introduction

show $\forall x.\ P(x)$
proof
  fix x local fixed variable
  show $P(x)$ $\langle proof \rangle$

show $\exists x.\ P(x)$
proof
  ⋮
  show $P(witness)$ $\langle proof \rangle$
qed

∀ and ∃ introduction

show $\forall x.\ P(x)$
proof
  fix x local fixed variable
  show $P(x)$ $\langle proof \rangle$

show $\exists x.\ P(x)$
proof
  ⋮
  show $P(witness)$ $\langle proof \rangle$
qed

Chains of equations

Textbook proof

$t_1 = t_2$ ⟨justification⟩
$\quad = t_3$ ⟨justification⟩
⋮
$\quad = t_n$ ⟨justification⟩

In Isabelle:

have $t_1 = t_2$ $\langle proof \rangle$
also have ... $= t_3$ $\langle proof \rangle$
⋮
also have ... $= t_n$ $\langle proof \rangle$

Chains of equations

Textbook proof

$t_1 = t_2$ ⟨justification⟩
$\quad = t_3$ ⟨justification⟩
⋮
$\quad = t_n$ ⟨justification⟩

In Isabelle:

have $t_1 = t_2$ $\langle proof \rangle$
also have ... $= t_3$ $\langle proof \rangle$
⋮
also have ... $= t_n$ $\langle proof \rangle$
finally show $t_1 = t_n$.

Chains of equations and inequations

Instead of $=$ you may also use $\leq$ and $<$.

Example

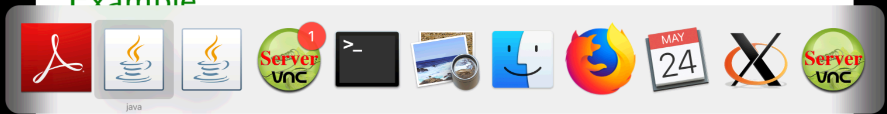
have $t_1 < t_2$ $\langle proof \rangle$
also have ... $= t_3$ $\langle proof \rangle$
$\vdots$
also have ... $\leq t_n$ $\langle proof \rangle$
finally show $t_1 < t_n$.

How to interpret "..."

Chains of equations and inequations

Instead of $=$ you may also use $\leq$ and $<$.

Example



$\vdots$
also have ... $\leq t_n$ $\langle proof \rangle$
finally show $t_1 < t_n$.

Example: pattern matching

show $formula_1 \longleftrightarrow formula_2$ (**is** $?L \longleftrightarrow ?R$)

Example: pattern matching

show $formula_1 \longleftrightarrow formula_2$ (**is** $?L \longleftrightarrow ?R$)
proof
  assume $?L$
  ⋮
  show $?R$ $\langle proof \rangle$
next
  assume $?R$
  ⋮
  show $?L$ $\langle proof \rangle$
qed

?thesis

show *formula*
proof -
  ⋮
  show *?thesis* $\langle proof \rangle$
qed

?thesis

show *formula* (*is ?thesis*)
proof -
  ⋮
  show *?thesis* $\langle proof \rangle$
qed

Every show implicitly defines *?thesis*

let

Introducing local abbreviations in proofs:

let $?t$ = "*some-big-term*"
⋮
have "... $?t$ ... "

Quoting facts by value

By name:

have *x0*: $''x > 0''$ ...
⋮
from *x0* ...

Quoting facts by value

By name:

have *x0*: $''x > 0''$ ...
⋮
from *x0* ...

By value:

have $''x > 0''$ ...
⋮
from ‘$x{>}0$‘ ...

Quoting facts by value

By name:

have *x0*: $''x > 0''$ ...
⋮
from *x0* ...

By value:

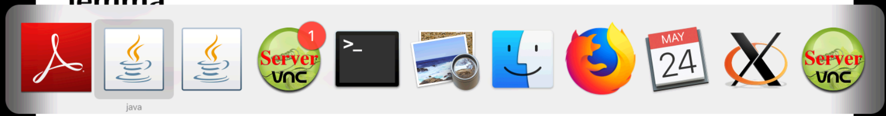
have $''x > 0''$ ...
⋮
from ‘$x{>}0$‘ ...
↑ ↑
back quotes

Example

lemma
$\exists\, ys\ zs.\ xs = ys\ @\ zs\ \wedge$
$(length\ ys = length\ zs \vee length\ ys = length\ zs + 1)$

Example

lemma



Local lemmas

have B **if** *name:* $A_1 \ \dots \ A_m$ **for** $x_1 \ \dots \ x_n$
$\langle proof \rangle$

Local lemmas

have B **if** *name:* $A_1 \ \dots \ A_m$ **for** $x_1 \ \dots \ x_n$
$\langle proof \rangle$

proves $\llbracket A_1; \dots ; A_m \rrbracket \Longrightarrow B$
where all x_i have been replaced by $?x_i$.

Proof state and Isar text

Proof state and Isar text

In general: **proof** *method*

Applies *method* and generates subgoal(s):

$$\bigwedge x_1 \ \dots \ x_n.\ [\![\ A_1;\ \dots\ ;\ A_m\ ]\!] \Longrightarrow B$$

Proof state and Isar text

In general: **proof** *method*

Applies *method* and generates subgoal(s):

$$\bigwedge x_1 \ \dots \ x_n.\ [\![\ A_1;\ \dots\ ;\ A_m\ ]\!] \Longrightarrow B$$

How to prove each subgoal:

fix $x_1 \ \dots \ x_n$
assume $A_1 \ \dots \ A_m$
⋮
show B

8 Isar by example

9 Proof patterns

10 Streamlining Proofs

11 Proof by Cases and Induction

Isar_Induction_Demo.thy

Proof by cases

Datatype case analysis

datatype $t \ = \ C_1 \ \vec{\tau} \ \mid \ \ldots$

proof $(cases \ "term")$
  case $(C_1 \ x_1 \ \ldots \ x_k)$
  $\ldots \ x_j \ \ldots$
next
$\vdots$
qed

Datatype case analysis

datatype $t \ = \ C_1 \ \vec{\tau} \ \mid \ \ldots$

proof $(cases \ "term")$
  case $(C_1 \ x_1 \ \ldots \ x_k)$
  $\ldots \ x_j \ \ldots$
next
$\vdots$
qed

where **case** $(C_i \ x_1 \ \ldots \ x_k) \quad \equiv$

fix $x_1 \ \ldots \ x_k$
assume $\underbrace{C_i\colon}_{\text{label}} \ \underbrace{term = (C_i \ x_1 \ \ldots \ x_k)}_{\text{formula}}$

Datatype case analysis

datatype $t \ = \ C_1 \ \vec{\tau} \ \mid \ \ldots$

proof $(cases \ "term")$
  case $(C_1 \ x_1 \ \ldots \ x_k)$
qed

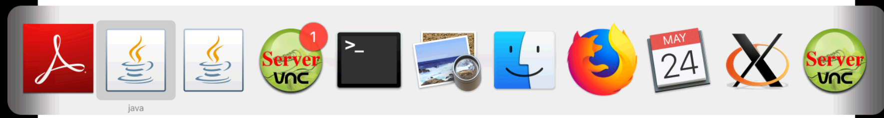
where **case** $(C_i \ x_1 \ \ldots \ x_k) \quad \equiv$

fix $x_1 \ \ldots \ x_k$
assume $\underbrace{C_i\colon}_{\text{label}} \ \underbrace{term = (C_i \ x_1 \ \ldots \ x_k)}_{\text{formula}}$

Isar_Induction_Demo.thy

Structural induction for *nat*

Isar_Induction_Demo.thy



Structural induction for *nat*

show $P(n)$
proof $(induction\ n)$
  case 0 $\equiv$ **let** $?case = P(0)$
  $\vdots$
  show $?case$
next
  case $(Suc\ n)$ $\equiv$ **fix** n **assume** Suc: $P(n)$
                    let $?case = P(Suc\ n)$
  $\vdots$
  $\vdots$
  show $?case$
qed

Structural induction with $\Longrightarrow$

show $A(n) \Longrightarrow P(n)$
proof $(induction\ n)$
  case 0
  $\vdots$
  show $?case$
next
  case $(Suc\ n)$
  $\vdots$

  $\vdots$
  show $?case$
qed

Structural induction with $\Longrightarrow$

show $A(n) \Longrightarrow P(n)$
proof $(induction\ n)$
  case 0 $\equiv$ **assume** 0: $A(0)$
  $\vdots$ **let** $?case = P(0)$
  show $?case$
next
  case $(Suc\ n)$
  $\vdots$

  $\vdots$
  show $?case$
qed

Structural induction with $\Longrightarrow$

show $A(n) \Longrightarrow P(n)$
proof $(induction\ n)$
  case 0 $\equiv$ **assume** 0: $A(0)$
  $\vdots$ **let** $?case = P(0)$
  show $?case$
next
  case $(Suc\ n)$ $\equiv$ **fix** n
  $\vdots$ **assume** Suc: $A(n) \Longrightarrow P(n)$
    $A(Suc\ n)$
  $\vdots$ **let** $?case = P(Suc\ n)$
  show $?case$
qed

Named assumptions

In a proof of

$$A_1 \Longrightarrow \ldots \Longrightarrow A_n \Longrightarrow B$$

by structural induction:

In the context of
  case C

we have
  $C.IH$ the induction hypotheses

Structural induction with $\Longrightarrow$

show $A(n) \Longrightarrow P(n)$
proof $(induction\ n)$
  case 0 $\equiv$ **assume** 0: $A(0)$
  $\vdots$ **let** $?case = P(0)$
  show $?case$
next
  case $(Suc\ n)$ $\equiv$ **fix** n
  $\vdots$ **assume** Suc: $A(n) \Longrightarrow P(n)$
    $A(Suc\ n)$
  $\vdots$ **let** $?case = P(Suc\ n)$
  show $?case$
qed

175

Named assumptions

In a proof of

$A_1 \Longrightarrow \ldots \Longrightarrow A_n \Longrightarrow B$

by structural induction:

In the context of

case C

we have

$C.IH$	the induction hypotheses
$C.prems$	the premises A_i

Named assumptions

In a proof of

$A_1 \Longrightarrow \ldots \Longrightarrow A_n \Longrightarrow B$

by structural induction:

In the context of

case C

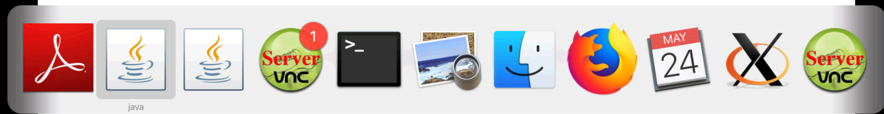
we have

$C.IH$	the induction hypotheses
$C.prems$	the premises A_i
C	$C.IH + C.prems$

`Isar_Induction_Demo.thy`

Computation induction

`Isar_Induction_Demo.thy`



Computation induction

Naming

- i is a name, but not $i.IH$
- Needs double quotes: "$i.IH$"
- Indexing: $i(1)$ and "$i.IH$"(1)
- If defining equations for f overlap:
  $\leadsto$ Isabelle instantiates overlapping equations
  $\leadsto$ case names of the form "i_j"