Доказательство теорем в системе Isabelle / HOL. Интенсивный курс.

- Данный курс подготовлен на основе курса формальных методов за авторством Тобиаса Нипкова (Tobias Nipkow), профессора Мюнхенского Технологического Университета.
- Note: this material is based on the original course developed by Prof. Tobias Nipkow from TUM University. For more details see http://isabelle.in.tum.de/coursematerial/PSV2009-1/

Примечание*

Переведено и озвучено:

- Якимов И.А.
  - ivan.yakimov.research@yandex.ru
- Кузнецов А.С.
  - askuznetsov@sfu-kras.ru

Слайды, отмеченные звездочкой*, добавлены переводчиками, так же добавлены некоторые примечания.

На момент перевода и чтения курса в СФУ оригинальный курс отрыт и доступен публично http://isabelle.in.tum.de/coursematerial/PSV2009-1/

Isar — язык структурированных доказательств

Apply сценарии

- Трудно читать
- Сложно сопровождать
- Не поддаются масштабированию

apply VS isar

apply = ассемблер

Isar = высокоуровневый язык

Но: apply полезны для исследований доказательств

Типичное доказательство в Isar

```
proof
  assume formula0
  have formula_1 by simp
  ...
  have formula_n by blast
  show formula_k by smth
qed
```

Проводит доказательство что из formula0 следует formula_k

Обзор

- Основные понятия
- Примеры
- Шаблоны доказательств

Базовый синтаксис

proof = **proof** [method] statement* **qed**
| **by** method

method = *(simp ...)* | *(blast ...)* | *(rule ...)* | ...

statement = **fix** variables ($\bigwedge$)
| **assume** prop ($\Longrightarrow$)
| [**from** fact$^+$] (**have** | **show**) prop proof
| **next** (separates subgoals)

prop = [name:] "formula"

fact = name | name**[**OF fact$^+$**]** | ‘formula‘

Пример

Теорема Кантора

We show a number of proofs of Cantor's theorem that a function from a set to its powerset cannot be surjective, illustrating various features of Isar. The constant *surj* is predefined.

```
lemma "¬ surj(f :: 'a ⇒ 'a set)"
proof (*assume surj, show False*)
 assume 0: "surj f"
 from 0 have 1: "∀y. ∃x. y = f x" by (simp add: surj_def)
 from 1 have 2: "∃b. {a. a ∉ f a} = f b" by auto
 from 2 show "False" by auto
qed
```

*Доказательство отпротивного по-умолчанию

The **proof** command lacks an explicit method by which to perform the proof. In such cases Isabelle tries to use some standard introduction rule, in the above case for $\neg$:

$$\frac{P \Longrightarrow \mathit{False}}{\neg\ P}$$

In order to prove $\neg\ P$, assume P and show False. Thus we may assume $\mathit{surj}\ f$. The proof shows that names of propositions may be (single!) digits — meaningful names are hard to invent and are often not necessary. Both **have** steps are obvious. The second one introduces the diagonal set $\{x.\ x \notin f\ x\}$, the key idea in the proof. If you wonder why 2 directly implies False: from 2 it follows that $(a \notin f\ a) = (a \in f\ a)$.

Аббривеатуры

- this = предыдущее утверждение, доказанное или выдвинутое
- then = from this
- thus = then show
- hence = then have

using

- Сначала что, затем как:

(**have**|**show**) prop **using** facts

=

from facts (**have**|**show**) prop

Пример: структурированное доказательство

```
lemma
 fixes f :: "'a ⇒ 'a set"
 assumes s: "surj f"
 shows "False"
proof - (* no automatic proof steps *)
 have "∃b. {a. a ∉ f a} = f b" using s
  by (auto simp: surj_def)
 thus "False" by blast
qed
```

*Структурированные леммы

fixes $x :: \tau_1$ **and** $y :: \tau_2 \ldots$
assumes a: P **and** b: $Q \ldots$
shows R

The optional **fixes** part allows you to state the types of variables up front rather than by decorating one of their occurrences in the formula with a type constraint. The key advantage of the structured format is the **assumes** part that allows you to name each assumption; multiple assumptions can be separated by **and**. The **shows** part gives the goal. The actual theorem that will come out of the proof is $surj\ f \Longrightarrow False$, but during the proof the assumption $surj\ f$ is available under the name s like any other fact.

Соль структурированных доказательств

- Посылки и промежуточные факты могут быть проименованы и далее использованы выборочно и в явном виде

Шаблоны доказательств

We show a number of important basic proof patterns. Many of them arise from the rules of natural deduction that are applied by **proof** by default. The patterns are phrased in terms of **show** but work for **have** and **lemma**, too.

We start with two forms of **case analysis**: starting from a formula P we have the two cases P and $\neg P$, and starting from a fact $P \vee Q$ we have the two cases P and Q:

Перебор вариантов и дизъюнкция

```
show "R"
proof cases
  assume "P"
   ⋮
 show "R" ...
next
  assume "¬ P"
   ⋮
 show "R" ...
qed
```

```
have "P ∨ Q" ...
then show "R"
proof
  assume "P"
   ⋮
 show "R" ...
next
  assume "Q"
   ⋮
 show "R" ...
qed
```

Логическая эквивалентность

$$
\begin{array}{l}
\textsf{show } "P \longleftrightarrow Q" \\
\textsf{proof} \\
\quad \textsf{assume } "P" \\
\qquad \vdots \\
\quad \textsf{show } "Q" \;\; \ldots \\
\textsf{next} \\
\quad \textsf{assume } "Q" \\
\qquad \vdots \\
\quad \textsf{show } "P" \;\; \ldots \\
\textsf{qed}
\end{array}
$$

От противного

```
show "¬ P"
proof
  assume "P"
    ⋮
  show "False" ...
qed
```

```
show "P"
proof (rule ccontr)
  assume "¬P"
    ⋮
  show "False" ...
qed
```

Квантификаторы

show $''\forall x.\ P(x)''$
proof
  fix x
  $\vdots$
  show $''P(x)''$ $\ldots$
qed

show $''\exists x.\ P(x)''$
proof
  $\vdots$
  show $''P(witness)''$ $\ldots$
qed

Дальнейшее чтение

- prog-prove, раздел Isar