http://coq.inria.fr/

http://coq.inria.fr/

http://lpcs.math.msu.su/~sk/

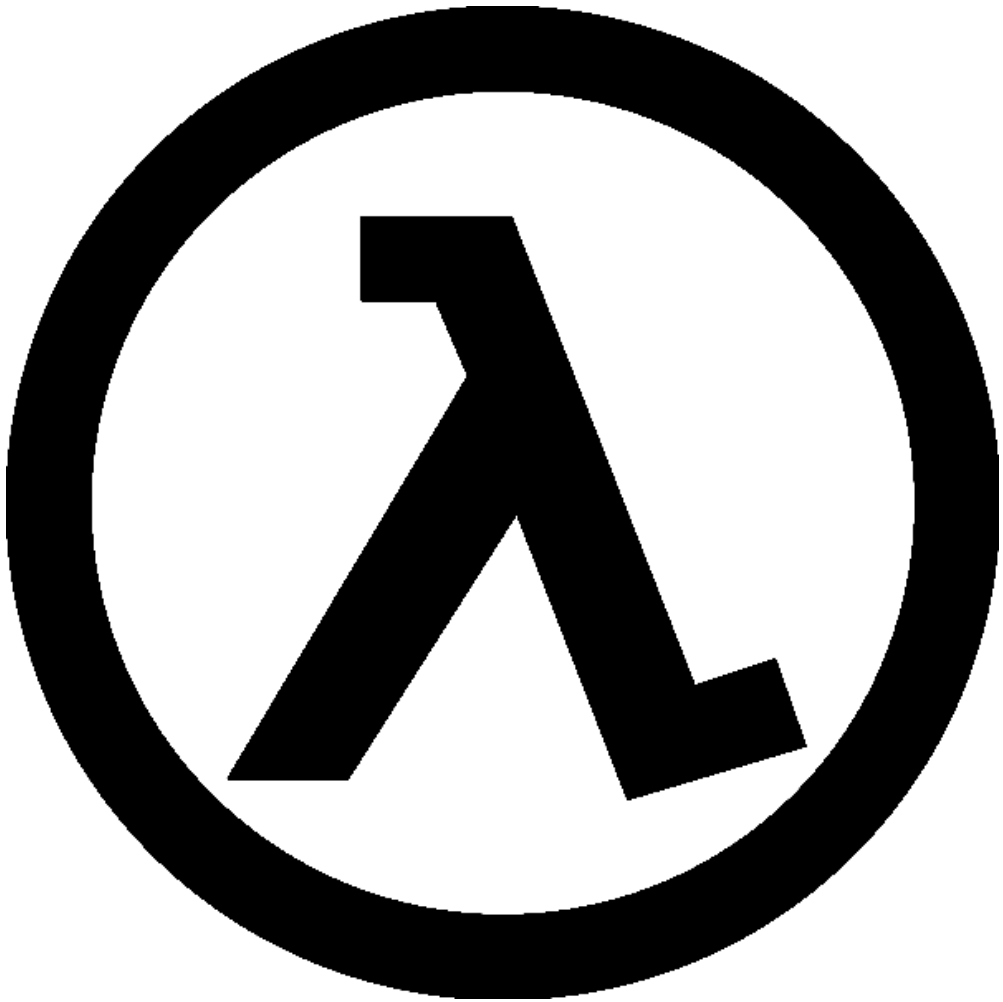
http://coq.inria.fr/

http://lpcs.math.msu.su/~sk/

sk@mi.ras.ru

λ-исчисление

λ-исчисление



«Парадокс»

Известно, что $(x^2)' = 2x$. Подставим $x = 1$; получим: $(1^2)' = 2 \cdot 1 = 2$. С другой стороны, $(1^2)' = 1' = 0$. Противоречие.

«Парадокс»

Известно, что $(x^2)' = 2x$. Подставим $x = 1$; получим: $(1^2)' = 2 \cdot 1 = 2$. С другой стороны, $(1^2)' = 1' = 0$. Противоречие.

Чтобы избежать подобной ошибки, нужно (как и в программировании!) внимательно следить за *типами* объектов. В данном случае сначала x^2 понимается как *функция* из $\mathbb{R}$ в $\mathbb{R}$ (объект типа $\mathbb{R} \to \mathbb{R}$), а потом — при подстановке $x = 1$ — просто как *действительное число* (объект типа $\mathbb{R}$).

Типы и термы

Обычно математики используют запись x^2 и для самой функции $\mathrm{sq}\colon x \mapsto x^2$, и для её *значения* в точке x, в зависимости от контекста, однако аккуратнее будет использовать более точную запись.

Типы и термы

Обычно математики используют запись x^2 и для самой функции $\mathrm{sq}\colon x \mapsto x^2$, и для её *значения* в точке x, в зависимости от контекста, однако аккуратнее будет использовать более точную запись.
Записи вида x^2, $2x$, $3x^3 - 4y$ оставим для *термов* типа «действительное число» (здесь x, y, ... — свободные переменные), а функцию sq будем обозначать так: $\lambda x.x^2$.

Типы и термы

Обычно математики используют запись x^2 и для самой функции $\mathrm{sq}\colon x \mapsto x^2$, и для её *значения* в точке x, в зависимости от контекста, однако аккуратнее будет использовать более точную запись.
Записи вида x^2, $2x$, $3x^3 - 4y$ оставим для *термов* типа «действительное число» (здесь x, y, ... — свободные переменные), а функцию sq будем обозначать так: $\lambda x.x^2$.
Оператор «λ», как и кванторы, *связывает* переменную: терм $\lambda x.x^2$ уже *замкнутый*, т.е. не содержит свободных переменных и обозначает один вполне конкретный объект.

Типы и термы

Применять операцию дифференцирования («$'$») к терму x^2 нельзя (дифференцировать можно функцию, а не число), зато можно применить дифференциальный оператор — обозначим его D — к терму $\lambda x.x^2$: $D(\lambda x.x^2) = \lambda x.2x$. Поскольку в этом равенстве переменная x связана, подстановка вместо неё константы запрещена, и парадокс снимается.

Типы и термы

Применять операцию дифференцирования («$'$») к терму x^2 нельзя (дифференцировать можно функцию, а не число), зато можно применить дифференциальный оператор — обозначим его D — к терму $\lambda x.x^2$: $D(\lambda x.x^2) = \lambda x.2x$. Поскольку в этом равенстве переменная x связана, подстановка вместо неё константы запрещена, и парадокс снимается.

Вопрос на понимание: какого типа сам дифференциальный оператор D?

Типы и термы

Применять операцию дифференцирования («$'$») к терму x^2 нельзя (дифференцировать можно функцию, а не число), зато можно применить дифференциальный оператор — обозначим его D — к терму $\lambda x.x^2$: $D(\lambda x.x^2) = \lambda x.2x$. Поскольку в этом равенстве переменная x связана, подстановка вместо неё константы запрещена, и парадокс снимается.

Вопрос на понимание: какого типа сам дифференциальный оператор D?
Ответ: $D : (\mathbb{R} \to \mathbb{R}) \to (\mathbb{R} \to \mathbb{R})$.

Типы и термы

Применять операцию дифференцирования («$'$») к терму x^2 нельзя (дифференцировать можно функцию, а не число), зато можно применить дифференциальный оператор — обозначим его D — к терму $\lambda x.x^2$: $D(\lambda x.x^2) = \lambda x.2x$. Поскольку в этом равенстве переменная x связана, подстановка вместо неё константы запрещена, и парадокс снимается.

Вопрос на понимание: какого типа сам дифференциальный оператор D?
Ответ: $D : (\mathbb{R} \to \mathbb{R}) \to (\mathbb{R} \to \mathbb{R})$.

Итак, типы строятся из некоторого набора базовых типов (в нашем случае это был тип $\mathbb{R}$) с помощью операции «$\to$».

Формальное определение

Типы:

1. Базовые типы.
2. Если A и B — типы, то $(A \to B)$ — тип.

Формальное определение

Типы:

1. Базовые типы.
2. Если A и B — типы, то $(A \to B)$ — тип.

Отметим аналогию с определением формул логики высказываний — это неспроста.

Формальное определение

Типы:

1. Базовые типы.
2. Если A и B — типы, то $(A \to B)$ — тип.

Отметим аналогию с определением формул логики высказываний — это неспроста.

Термы: каждый терм имеет тип. Тот факт, что терм t имеет тип A, обозначим $t : A$.

1. Константы: для каждого типа A имеется счётное множество констант данного типа c_1^A, c_2^A, ...
2. Переменные: для каждого типа A имеется счётное множество переменных данного типа x_1^A, x_2^A, ...
3. *Произведение* (применение функции): если $t : A \to B$ и $s : A$, то $(t \cdot s)$ — терм типа B (если же типы термов t и s не согласованы, то выражение $(t \cdot s)$ не является термом).
4. *λ-абстракция:* если $t : B$, то $\lambda x^A.t$ — терм типа $A \to B$.

Соответствие Карри – Говарда

Theorem

Если A — тип и существует такой замкнутый *λ-терм u, что $u : A$. Тогда A, если рассмотреть его как формулу логики высказываний, будет* **интуиционистской** *тавтологией.*

$\mathrm{Int}_{\to}$ (система «естественного вывода»)

$$\Gamma, A \vdash A$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash (A \to B)}$$

$$\frac{\Gamma \vdash A \quad \Gamma \vdash (A \to B)}{\Gamma \vdash B}$$

$\mathrm{Int}_{\to}$ (система «естественного вывода»)

$$\Gamma, A \vdash A$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash (A \to B)} \qquad x : A, u : B \rightsquigarrow \lambda x.u : (A \to B)$$

$$\frac{\Gamma \vdash A \quad \Gamma \vdash (A \to B)}{\Gamma \vdash B} \qquad u : (A \to B), v : A \rightsquigarrow (uv) : B$$

Соответствие Карри – Говарда

$$x_1 : A_1, \ldots, x_n : A_n \vdash u(x_1, \ldots, x_n) : A$$

Соответствие Карри – Говарда

$$x_1 : A_1, \ldots, x_n : A_n \vdash u(x_1, \ldots, x_n) : A$$

Терм u **кодирует** вывод в $\mathrm{Int}_{\rightarrow}$.

Соответствие Карри – Говарда

$$x_1 : A_1, \ldots, x_n : A_n \vdash u(x_1, \ldots, x_n) : A$$

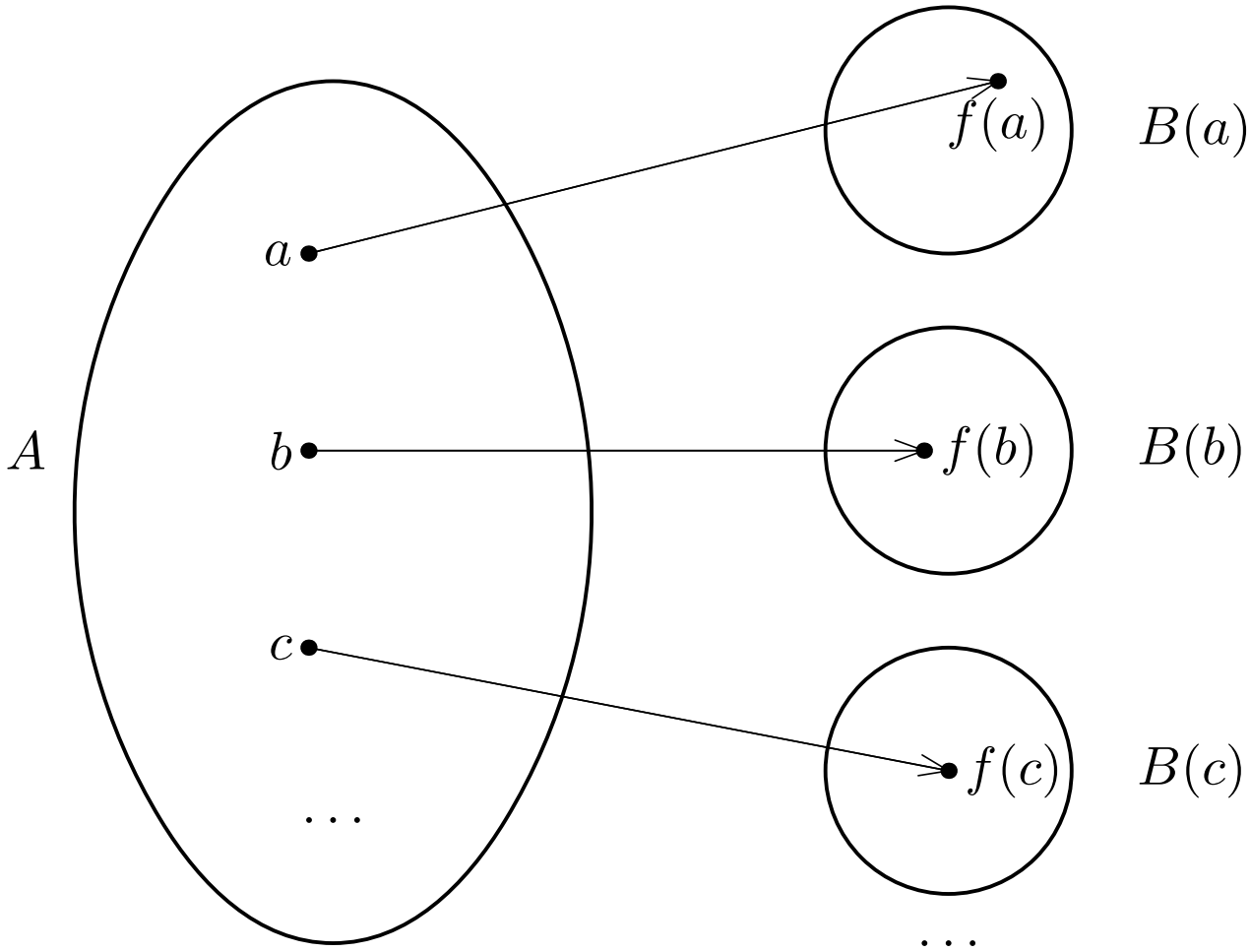
Терм u **кодирует** вывод в $\mathrm{Int}_{\to}$.

Закон Пирса. Не существует замкнутого λ-терма типа $((p \to q) \to p) \to p$.

“Propositions as Types” (логика 1-го порядка и зависимое произведение)

Тип	Высказывание
$A \to B$ населен функциями f, которые при при каждом $x : A$ определены и принимают значение $f(x) : B$.	$A \to B$ верно, если имеется конструкция f, которая каждое доказательство x высказывания A преобразует в доказательство $f(x)$ высказывания B.
$\Pi_{x:T} B(x)$ населен функциями f, которые при при каждом $x : T$ определены и принимают значение $f(x) : B(x)$.	$\forall x : T, B(x)$ верно, если имеется конструкция f, которая каждое значение x типа T преобразует в доказательство $f(x)$ высказывания $B(x)$.

Зависимое произведение типов



“Propositions as Types”

Тип	Высказывание
$A \times B$	$A \wedge B$
$A \oplus B$	$A \vee B$
$\sum_{x:T} B(x)$	$\exists x : T, B(x)$
$void$	$\bot$
$void \to void$	$\top$

“Propositions as Types”

Тип	Высказывание
$A \times B$	$A \wedge B$
$A \oplus B$	$A \vee B$
$\sum_{x:T} B(x)$	$\exists x : T, B(x)$
$void$	$\bot$
$void \to void$	$\top$

Доказательства — λ-термы соответствующих типов.

Секвенции

$$x_1 : A_1, x_2 : A_2(x_1), \ldots, x_n : A_n(x_1, \ldots, x_{n-1}) \vdash t(x_1, \ldots, x_n) : B(x_1, \ldots, x_n)$$

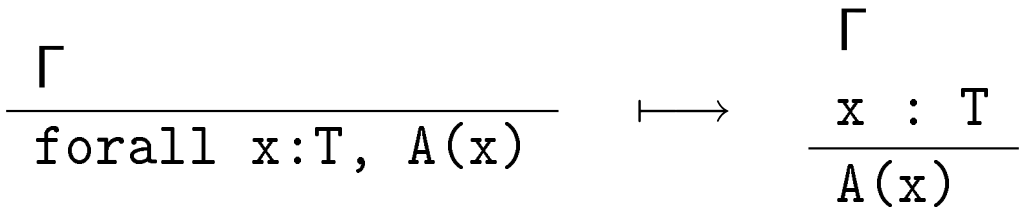
Секвенции

$$x_1 : A_1, x_2 : A_2(x_1), \ldots, x_n : A_n(x_1, \ldots, x_{n-1}) \vdash (\mathbf{???}) : B(x_1, \ldots, x_n)$$

Тактика assumption

$$\frac{\begin{array}{l}\Gamma \\ \mathtt{H} : \mathtt{A}\end{array}}{\mathtt{A}} \quad \longmapsto \quad \texttt{Proof completed}$$

Тактика intro



Тактика intro

$$\frac{\Gamma}{\texttt{forall x:T, A(x)}} \quad \longmapsto \quad \frac{\Gamma,\ \texttt{x : T}}{\texttt{A(x)}}$$

$$\frac{\Gamma}{\texttt{A -> B}} \quad \longmapsto \quad \frac{\Gamma,\ \texttt{H : A}}{\texttt{B}}$$

Тактика intro

$$\frac{\Gamma}{\texttt{forall x:T, A(x)}} \quad \longmapsto \quad \frac{\Gamma}{\texttt{x : T}}$$

$$\texttt{A(x)}$$

$$\frac{\Gamma}{\texttt{A -> B}} \quad \longmapsto \quad \frac{\Gamma}{\texttt{H : A}}$$

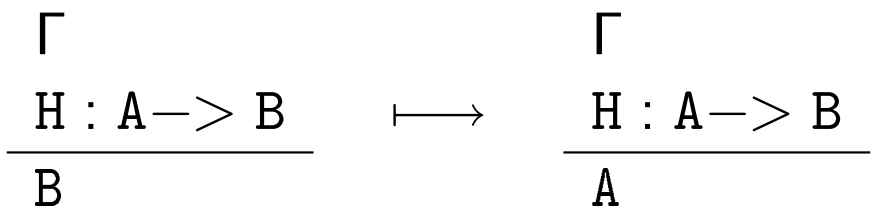
$$\texttt{B}$$

Искомый терм t получается из терма t_1 для подцели с помощью λ-абстракции: $t = \lambda x{:}T.\, t_1$ и $t = \lambda H{:}A.\, t_1$ соответственно. В синтаксисе системы Coq они записываются так: $t = (\texttt{fun x : T} \Rightarrow t_1)$ и $t = (\texttt{fun H : T} \Rightarrow t_1)$.

Тактика intro: соответствующее логическое правило

$$\frac{\Gamma, x : T \vdash u(x) : A(x)}{\Gamma \vdash (\lambda x^T.u(x)) : \forall x^T\, A(x)}$$

Тактика apply



Тактика apply

$$\frac{\Gamma \quad H : A \to B}{B} \quad \longmapsto \quad \frac{\Gamma \quad H : A \to B}{A}$$

$$\frac{\Gamma \quad H : A \to B \to C}{C} \quad \longmapsto \quad \frac{\Gamma \quad H : A \to B \to C}{A} \quad , \quad \frac{\Gamma \quad H : A \to B \to C}{B}$$

Импликация

$$(\texttt{A} \texttt{->} \texttt{B}) = (\texttt{forall x : A, B}), \qquad \texttt{x} \notin FV(\texttt{B}).$$

Конъюнкция

```
Inductive and (A B : Prop) : Prop :=
conj : A -> B -> A ∧ B .
```

Конъюнкция

```
Inductive and (A B : Prop) : Prop :=
conj : A -> B -> A ∧ B .
```

Тактика `split` (`apply conj`):

$$\frac{\Gamma}{A \wedge B} \longmapsto \frac{\Gamma}{A}\ ,\ \frac{\Gamma}{B}\ .$$

Конъюнкция

```
Inductive and (A B : Prop) : Prop :=
conj : A -> B -> A ∧ B .
```

Тактика `split` (`apply conj`):

$$\frac{\Gamma}{A \wedge B} \quad \longmapsto \quad \frac{\Gamma}{A} \;, \quad \frac{\Gamma}{B} \quad .$$

Тактика `elim`.

$$\frac{\begin{array}{l}\Gamma \\ H : A \wedge B\end{array}}{C} \quad \longmapsto \quad \frac{\Gamma}{A \to B \to C} \quad .$$

Дизъюнкция

```
Inductive or (A B : Prop) : Prop :=
or_introl : A -> A ∨ B
| or_intror : B -> A ∨ B .
```

Тактики `left` и `right`:

$$\frac{\Gamma}{A \vee B} \longmapsto \frac{\Gamma}{A} \quad \text{и} \quad \frac{\Gamma}{A \vee B} \longmapsto \frac{\Gamma}{B}$$

Тактика `elim`:

$$\frac{\begin{array}{l}\Gamma \\ H : A \vee B\end{array}}{C} \longmapsto \frac{\begin{array}{l}\Gamma \\ H : A \vee B\end{array}}{A \to C}, \quad \frac{\begin{array}{l}\Gamma \\ H : A \vee B\end{array}}{B \to C}$$

Константы True и False, отрицание

```
Inductive False : Prop := .
```

Константы True и False, отрицание

```
Inductive False : Prop := .

Inductive True : Prop := I : True .
```

Константы True и False, отрицание

```
Inductive False : Prop := .

Inductive True : Prop := I : True .

not = fun A : Prop => A -> False : Prop -> Prop
```

Константы True и False, отрицание

```
Inductive False : Prop := .

Inductive True : Prop := I : True .

not = fun A : Prop => A -> False : Prop -> Prop
```

Раскрытие отрицания — тактика **red** (преобразование по определению в импликацию ко лжи).

Тактика contradict

$$\frac{\Gamma \quad H : \sim A}{B} \longmapsto \frac{\Gamma}{A} \qquad \frac{\Gamma \quad H : A}{B} \longmapsto \frac{\Gamma}{\sim A}$$

$$\frac{\Gamma \quad H : \sim A}{\sim B} \longmapsto \frac{\Gamma \quad H : B}{A} \qquad \frac{\Gamma \quad H : A}{\sim B} \longmapsto \frac{\Gamma \quad H : B}{\sim A}$$

Классическая логика

Закон исключённого третьего:

```
Axiom classic: forall P : Prop, P \/ ~P.
```

Классическая логика

Закон исключённого третьего:

```
Axiom classic: forall P : Prop, P \/ ~P.
```

```
Theorem NNPP: forall p : Prop, ~~p -> p.
Theorem proof_irrelevance: forall (P : Prop)(p1 p2 :
P), p1 = p2.
```

Классическая логика

Закон исключённого третьего:

```
Axiom classic: forall P : Prop, P \/ ~P.
```

```
Theorem NNPP: forall p : Prop, ~~p -> p.
Theorem proof_irrelevance: forall (P : Prop)(p1 p2 :
P), p1 = p2.
```

Модуль: `Require Import Classical.`

(Интуиционистский) квантор существования

```
Inductive ex (A : Type) (P : A -> Prop) : Prop :=
   ex_intro : forall x : A, P x -> ex P.
```

(Интуиционистский) квантор существования

```
Inductive ex (A : Type) (P : A -> Prop) : Prop :=
   ex_intro : forall x : A, P x -> ex P.
```

Команда “`exists a.`”

$$\frac{\Gamma}{\texttt{exists x : A, P x}} \quad \longmapsto \quad \frac{\Gamma}{\texttt{P a}}$$

(Интуиционистский) квантор существования

```
Inductive ex (A : Type) (P : A -> Prop) : Prop :=
    ex_intro : forall x : A, P x -> ex P.
```

Команда “exists a.”

$$\frac{\Gamma}{\texttt{exists x : A, P x}} \quad \longmapsto \quad \frac{\Gamma}{\texttt{P a}}$$

Команда “elim H.”

$$\frac{\begin{array}{l}\Gamma \\ \texttt{H : exists x : A, P x}\end{array}}{\texttt{B}} \quad \longmapsto \quad \frac{\begin{array}{l}\Gamma \\ \texttt{H : exists x : A, P x}\end{array}}{\texttt{forall x : A, P x -> B}}$$

Сечение

Правило сечения

$$\frac{\Gamma \vdash A \quad \Gamma, A \vdash B}{\Gamma \vdash B}$$

реализовано в виде тактики `assert`, которой в качестве параметра надо передавать высекаемую формулу (команда “`assert A.`”).

$$\frac{\Gamma}{\mathtt{B}} \quad \longmapsto \quad \frac{\Gamma}{\mathtt{A}} \quad , \quad \frac{\begin{array}{c}\Gamma \\ \mathtt{H}:\ \mathtt{A}\end{array}}{\mathtt{B}} \ .$$

Сечение

Правило сечения

$$\frac{\Gamma \vdash A \quad \Gamma, A \vdash B}{\Gamma \vdash B}$$

реализовано в виде тактики `assert`, которой в качестве параметра надо передавать высекаемую формулу (команда "`assert A.`").

$$\frac{\Gamma}{\mathtt{B}} \longmapsto \frac{\Gamma}{\mathtt{A}}\,, \quad \frac{\begin{array}{c}\Gamma \\ \mathtt{H:\ A}\end{array}}{\mathtt{B}}\,.$$

Близкий эффект достигается также командой "`cut A.`", реализующей правило Modus Ponens:

$$\frac{\Gamma}{\mathtt{B}} \longmapsto \frac{\Gamma}{\mathtt{A -> B}}\,, \quad \frac{\Gamma}{\mathtt{A}}\,.$$