

Interpretacje symboli, język taktyk

12 maja 2009

Plan

- Notacje i zakresy interpretacji symboli
- Język taktyk Ltac

- Polecenie Notation
 - standardowe „poziomy” (RefMan, fig. 3.1, p. 84)
- Zakresy interpretacji (*interpretation scopes*)
 - definiowanie (w locie)
 - otwieranie i zamykanie
 - stos interpretacji
 - używanie na chwilę (*term*)%klucz

Uwaga: słowa kluczowe są globalne!

- Wiązanie zakresów z typami
 - `explicite: Arguments Scope ...`
 - `implicite: Bind Scope ... with ...`
- Odkrywanie notacji: `Locate, Print Grammar constr`
- Standardowe zakresy interpretacji (RefMan Sect 12.2.4, p. 292)
 - stałe liczbowe i napisowe (Ocaml :)
 - symbole
 - notacje specjalne

Język taktyk (operatory)

- proste taktyki złożone (*tacticals*):

```
destruct H; auto
destruct H; [ auto | idtac | apply H1; eauto ]
destruct H; [ auto | | apply H1; eauto ]
destruct H; [ auto | .. ]
destruct H; [ auto | eauto.. ]
apply H0 || apply H1 (* pierwsza, która coś zdołała *)
first [ apply H0 | apply H1 | apply H2 ]
      (* pierwsza, która nie padnie *)
solve [ apply H0 | apply H1 | apply H2 ]
      (* pierwsza, która rozwiąże *)
repeat (rewrite H; simpl)
do 5 rewrite H
progress auto
try tauto
fail
```

Język taktyk: konstrukcje

- definicje taktyk:

```
Ltac taktyka := intuition (subst; eauto).
```

```
Ltac tak n := intro n; elim n.
```

- analiza:

```
* match t with ... end
```

```
* match [reverse] goal with
```

```
  | |- t                => tac
```

```
  | H:cos |- cos       => tac ...
```

```
  | ... C[?X]          => ... context C[term]
```

```
end
```

```
* lazymatch ...
```

- rekurencja

Język taktyk: konstrukcje (c.d)

- generowanie nazw i termow

```
fresh
fresh n
type of t
eval compute in t
```

- reszta

```
fun ... => ...
let x:=tac in tac
let rec x:=tac in tac
abstract ...
```

- semantyka: taktyki, termy, (liczby, napisy)

```
constr: term
```