Institut National
Universitaire
Champollion

Types de données, preuves

Cours-TD n° 2

Chapitre 2 - Récurrence et types récursifs

Mélanger du raisonnement par récurrence et des types récursifs (Caml)

Laura Brillon

Sommaire

Exemple introductif : l'arithmétique de Peano

G. Peano a proposé une définition axiomatique de l'arithmétique (1889).

Il a défini les entiers naturels à l'aide d'un type somme récursif :

```
type ent_nat =
Zero
| Succ of ent_nat ;;
```

Exemple : La représentation de l'entier 3 est alors

`Succ(Succ(Succ(Zero)))`

Arithmétique de Peano

Comme peut-on faire une "récurrence" pour démontrer une propriété de l'arithmétique de Peano ?

↬ Par *récurrence structurelle*

Pour montrer $\forall n, P(n)$ est vraie

Initialisation : Montrer que $P(Zero)$ est vraie

Hérédité : Montrer que $P(n) \Rightarrow P(Succ(n))$
c'est à dire, supposer que $P(n)$ est vraie et montrer que $P(Succ(n))$ l'est également.

Exercice : Addition dans l'arithmétique de Peano

Sommaire

Les listes

Les listes sont aussi un type récursif !

Les constructeurs sont :

- La liste vide : `[ ] : 'a list`
- L'opérateur "Cons" : `(::) : 'a -> 'a list -> 'a list`

Exemple : La liste `[2;3]` se représente

`2 :: 3 :: [ ]`

Exemple

Exemple

```
let rec long = function
 [ ] -> 0
| x :: liste -> 1 + long
liste ;;
```

```
long 2::3::[ ] ->      2
   long 3::[ ] ->   1
      long [ ] -> 0
```

Comment prouver que `long liste` $\geq$ 0 pour toute liste `liste` ?

↬ Une preuve par récurrence structurelle permet de montrer qu'une assertion est vraie sur *tous les éléments d'un type récursif* en "reconstituant" le processus de construction des éléments.

Schéma de récurrence structurelle sur les listes

Schéma de récurrence structurelle sur les listes

Pour montrer $\forall \mathit{liste}, P(\mathit{liste})$ est vraie :

- **Initialisation :** Montrer que $P([\ ])$ est vraie
- **Hérédité :** Montrer que $P(\mathit{liste}) \Rightarrow P(x :: \mathit{liste})$ est vraie pour tout élément `x` et toute liste `liste`
  c'est à dire, supposer que $P(\mathit{liste})$ est vraie pour une liste liste quelconque, et montrer que pour tout élément x, $P(x :: \mathit{liste})$ est encore vraie.

Exemple : Montrer que `long list` $\geq$ 0 pour tout liste `liste` par récurrence structurelle.

Autres types récursifs

Autres types récursifs

Schéma de récurrence

Pour tout type récursif `T`, nous pouvons dériver un *schéma de récurrence* de manière similaire.

Pour montrer $\forall t : \texttt{T},\ P(t)$

- **Initialisation :** Pour tout constructeur de base C_b `of` $T_1 * \ldots * T_n$, montrer

$$\forall x_1, \ldots, x_n, P(C_b(x_1, \ldots, x_n))$$

- **Hérédité :** Pour tout constructeur récursif C_i `of` $T * \ldots * T * T_1 * \ldots * T_m$, montrer

$$\forall x_1, \ldots, x_m, P(t_1) \wedge \ldots \wedge P(t_k) \rightarrow P(C_i(t_1, \ldots, t_k, x_1, \ldots, x_m))$$

Exemple : Arbres binaires

Exemple

Arbres binaires :

```
type 'a arbre_bin =
Feuille of 'a
| Noeud of 'a * 'a arbre_bin * 'a arbre_bin ;;
```

Schéma de récurrence :

Pour montrer $\forall t$: `'a arbre_bin`, $P(t)$:

- Montrer $\forall \ell, P(\mathit{Feuille}\ \ell)$
- Montrer $\forall n, P(t_1) \wedge P(t_2) \to P(\mathit{Noeud}(n, t_1, t_2))$