

Preuve de propriétés de programmes

Récurrence :

si $S(0) \wedge \forall n S(n) \Rightarrow S(n+1)$

alors $\forall m S(m)$

Récurrence *structurelle* sur des données :

si $P(\text{nil}) \wedge \forall a, L \in \text{listes } P(L) \Rightarrow P(\text{cons } a L)$

alors $\forall m \in \text{listes } P(m)$

Quelques axiomes

$$\text{nil} \neq (\text{cons } x \ y)$$

$$(\text{cons } x \ y) = (\text{cons } u \ v) \Leftrightarrow x = u \wedge y = v$$

$$(\text{car } \text{nil}) = \text{nil}$$

$$(\text{cdr } \text{nil}) = \text{nil}$$

$$(\text{null } \text{nil}) = (\text{cons } \text{nil } \text{nil}) = \text{t}$$

$$(\text{null } (\text{cons } x \ y)) = \text{nil}$$

$$(\text{car } (\text{cons } x \ y)) = x$$

$$(\text{cdr } (\text{cons } x \ y)) = y$$

$$(\text{equal } x \ y) = \text{nil} \Leftrightarrow x \neq y$$

$$(\text{equal } x \ y) = \text{t} \Leftrightarrow x = y$$

$$(\text{if } (\text{cons } x \ y) \ u \ v) = u$$

$$(\text{if } \text{nil} \ u \ v) = v$$

Et quelques normalisations :

$$(\text{if } x \ x \ \text{nil}) = x$$

$$(\text{if } x \ y \ y) = y$$

....

Schéma de preuve

définition : le programme

théorème but : la spécification $P =$ l'hypothèse de récurrence

preuve : récurrence sur une variable v décomposable

(problème de son choix)

base : P avec $v = \text{nil}$

exécution symbolique tant que possible : SIMPLIFICATION

si l'égalité est satisfaite OK

sinon : prendre l'état courant de l'égalité, la poser comme sous-théorème but :

LEMME et faire une **preuve** dessus

pas : P avec $v = (\text{cons } e \ v)$

exécution symbolique

+ substitution de l'hypothèse de récurrence : CROSS-FERTILISATION

+ GENERALISATION

finPreuve

finThéorème

Algorithme de preuve (pour le pas)

Procedure recurrence (theorem)

loop:

oldTheorem := theorem

theorem := reduce (normalize (evalsym (theorem)))

si theorem = t **alors** exit

si theorem ≠ oldTheorem **alors goto** loop

si fertilisation est possible

alors theorem := fertilize (theorem)

sinon theorem := récurrence (generalise (theorem))

goto loop

Un premier exemple de preuve (1)

Avec la définition :

```
(de app (x y)
  (if (null x) y (cons (car x) (app (cdr x) y))))
```

Il faut démontrer l'hypothèse

$(\text{app } A (\text{app } B C)) = (\text{app } (\text{app } A B) C)$

Un premier exemple de preuve (2)

```
(de app (x y)
  (if (null x) y (cons (car x) (app (cdr x) y))))
```

1. Choix d'une variable décomposable : x

2. Preuve de la base :

$(\text{app nil } (\text{app B C})) = (\text{app } (\text{app nil B}) C)$

3. l'exécution symbolique de la partie gauche :

avec $x \leftarrow \text{nil}$ et $y \leftarrow (\text{app B C})$

donne :

$(\text{if } (\text{null nil}) (\text{app B C}) (\text{cons } (\text{car nil}) (\text{app } (\text{cdr nil}) (\text{app B C}))))$

$\rightarrow (\text{app B C})$

Un premier exemple de preuve (3)

```
(de app (x y)
        (if (null x) y (cons (car x) (app (cdr x) y))))
```

4. l'exécution symbolique de la partie droite (app (app nil B) C) :
avec : $x \leftarrow \text{nil}$ et $y \leftarrow B$

donne :

(app (if (null nil) B (cons (car nil) (app (cdr nil) B))) C)

→ (app B C)

5. Puisque :

simplification(partie gauche) = simplification (partie droite)

(app B C) = (app B C) ✓

la base est démontrée !

Un premier exemple de preuve (4)

```
(de app (x y)
      (if (null x) y (cons (car x) (app (cdr x) y))))
```

6. Ensuite il faut démontrer le pas :

$$(\text{app } (\text{cons } a \ A) \ (\text{app } B \ C)) = (\text{app } (\text{app } (\text{cons } a \ A) \ B) \ C)$$

7. L'exécution symbolique de la partie gauche

avec $x \leftarrow (\text{cons } a \ A)$ et $y \leftarrow (\text{app } B \ C)$

donne :

```
(if (null (cons a A)) (app B C)
```

```
    (cons (car (cons a A)) (app (cdr (cons a A)) (app B C))))
```

```
→ (cons (car (cons a A)) (app (cdr (cons a A)) (app B C)))
```

```
→ (cons a (app A (app B C)))
```

plus de simplification possible !

Un premier exemple de preuve (5)

```
(de app (x y)
        (if (null x) y (cons (car x) (app (cdr x) y))))
```

8. exécution symbolique de la partie droite $(\text{app } (\text{app } (\text{cons } a \ A) \ B) \ C)$
avec $x \leftarrow (\text{cons } a \ A)$ et $y \leftarrow B$

donne :

$(\text{app } (\text{if } (\text{null } (\text{cons } a \ A)) \ B$
 $\quad (\text{cons } (\text{car } (\text{cons } a \ A)) (\text{app } (\text{cdr } (\text{cons } a \ A)) \ B))) \ C)$

$\rightarrow (\text{app } (\text{cons } (\text{car } (\text{cons } a \ A)) (\text{app } (\text{cdr } (\text{cons } a \ A)) \ B)) \ C)$

$\rightarrow (\text{app } (\text{cons } a \ (\text{app } A \ B)) \ C)$

et avec $x \leftarrow (\text{cons } a \ (\text{app } A \ B))$ et $y \leftarrow C$

$\rightarrow (\text{if } (\text{null } (\text{cons } a \ (\text{app } A \ B))) \ C$

$\quad (\text{cons } (\text{car } (\text{cons } a \ (\text{app } A \ B)))$

$\quad (\text{app } (\text{cdr } (\text{cons } a \ (\text{app } A \ B))) \ C)))$

Un premier exemple de preuve (6)

```
(de app (x y)
  (if (null x) y (cons (car x) (app (cdr x) y))))
```

→ (cons (car (cons a (app A B))) (app (cdr (cons a (app A B))) C))

→ (cons a (app (app A B) C) plus de simplification possible !

nous avons maintenant :

(cons a (app A (app B C))) = (cons a (app (app A B) C))

et l'hypothèse :

(app A (app B C)) = (app (app A B) C)

9. Nous pouvons cross-fertiliser (suivant l'une ou l'autre flèche) :

(cons a (app (app A B) C)) = (cons a (app (app A B) C))

Identité ! Donc nous avons démontré que

$\forall A, B, C$ (app A (app B C)) = (app (app A B) C)

Deuxième exemple (1)

soit le programme **app** précédent et le programme **rev** ci-dessous:

```
(de rev (l)
  (if (null l) nil
      (app (rev (cdr l)) (cons (car l) nil))))
```

Il s'agit de démontrer l'hypthèse :

$$(\text{app } (\text{rev } A) (\text{rev } B)) = (\text{rev } (\text{app } B A))$$

Deuxième exemple (2)

(de rev (l) (if (null l) nil (app (rev (cdr l)) (cons (car l) nil))))

(de app (x y) (if (null x) y (cons (car x) (app (cdr x) y))))

Choix de la variable de récurrence : **B**

Base : (app (rev A)(rev nil)) = (rev (app nil A))

$l \leftarrow \text{nil}$ et $x \leftarrow \text{nil}$

(app (rev A) nil) = (rev A) et *blockage* !

Donc LEMME : démontrons (app (rev A) nil) = (rev A)

Et pourquoi pas le généraliser et démontrer le LEMME :

(app C nil) = C

Deuxième exemple (3)

```
(de rev (l) (if (null l) nil (app (rev (cdr l)) (cons (car l) nil))))  
(de app (x y) (if (null x) y (cons (car x) (app (cdr x) y))))
```

Base du Lemme $(\text{app } C \text{ nil}) = C$:

$(\text{app nil nil}) = \text{nil} \Rightarrow \text{nil} = \text{nil}$ **vrai !**

Pas du lemme :

$(\text{app (cons c C) nil}) = (\text{cons c C})$ ←

Partie gauche avec $x \leftarrow (\text{cons c C})$ et $y \leftarrow \text{nil}$:

$(\text{if (null (cons c C) nil (cons (car (cons c C))(app (cdr (cons c C) nil))))$
 $(\text{cons c (app C nil)})$

Fertilisation avec l'hypothèse $(\text{app } C \text{ nil}) = C$ donne :

(cons c C) ce qui est bien égal à la partie droite (cons c C)

Donc : $(\text{app } C \text{ nil}) = \text{nil}$ **démontré**,

donc : base de $(\text{app (rev A) (rev B)}) = (\text{rev (app B A)})$ **démontré**

Deuxième exemple (4)

```
(de rev (l) (if (null l) nil (app (rev (cdr l)) (cons (car l) nil))))  
(de app (x y) (if (null x) y (cons (car x) (app (cdr x) y))))
```

Le pas de l'hypothèse $(\text{app} (\text{rev } A) (\text{rev } B)) = (\text{rev} (\text{app } B A))$:

$(\text{app} (\text{rev } A) (\text{rev} (\text{cons } b B))) = (\text{rev} (\text{app} (\text{cons } b B) A))$

Exécution symbolique de la partie gauche donne :

$(\text{app} (\text{rev } A) (\text{app} (\text{rev} (\text{cdr} (\text{cons } b B))) (\text{cons} (\text{car} (\text{cons } b B)) \text{nil})))$

$(\text{app} (\text{rev } A) (\text{app} (\text{rev } B) (\text{cons } b \text{nil})))$

Exécution symbolique de la partie droite donne :

$(\text{rev} (\text{cons} (\text{car} (\text{cons } b B)) (\text{app} (\text{cdr} (\text{cons } b B)) A)))$

$(\text{rev} (\text{cons } b (\text{app } B A)))$

$(\text{app} (\text{rev} (\text{cdr} (\text{cons } b (\text{app } B A)))) (\text{cons} (\text{car} (\text{cons } b B)) \text{nil}))$

$(\text{app} (\text{rev} (\text{app } B A)) (\text{cons } b \text{nil}))$

Deuxième exemple (5)

```
(de rev (l) (if (null l) nil (app (rev (cdr l)) (cons (car l) nil))))  
(de app (x y) (if (null x) y (cons (car x) (app (cdr x) y))))
```

On a maintenant :

$(\text{app } (\text{rev } A) (\text{app } (\text{rev } B) (\text{cons } b \text{ nil})))$

$= (\text{app } (\text{rev } (\text{app } B A)) (\text{cons } b \text{ nil}))$

Une **cross-fertilisation** avec $(\text{rev } (\text{app } B A)) = (\text{app } (\text{rev } A) (\text{rev } B))$

donne pour la partie droite :

$= (\text{app } (\text{app } (\text{rev } A) (\text{rev } B)) (\text{cons } b \text{ nil}))$

Une **généralisation** s'impose : $(\text{rev } A) \mapsto X$, $(\text{rev } B) \mapsto Y$, $(\text{cons } b \text{ nil}) \mapsto Z$

Ce qui donne :

$(\text{app } X (\text{app } Y Z)) = (\text{app } (\text{app } X Y) Z)$

ce qui nous ramène au théorème précédemment démontré !

$$(\text{rev} (\text{rev} A)) = A \quad (1)$$

Toujours avec les deux mêmes fonctions rev et app démontrons
l'hypothèse que :

$$(\text{rev} (\text{rev} A)) = A$$

La base : $(\text{rev} (\text{rev} \text{nil})) = \text{nil}$

$$(\text{rev} \text{nil}) = \text{nil}$$

$$\text{nil} = \text{nil}$$

Le Pas : $(\text{rev} (\text{rev} (\text{cons} a A))) = (\text{cons} a A)$

$$(\text{rev} (\text{app} (\text{rev} A) (\text{cons} a \text{nil}))) = (\text{cons} a A)$$

Blocké ! Cross-fertilisation de la partie droite donne :

$$(\text{rev} (\text{app} (\text{rev} A) (\text{cons} a \text{nil}))) = (\text{cons} a (\text{rev} (\text{rev} A)))$$

$$(\text{rev} (\text{rev } A)) = A \quad (2)$$

```
(de rev (l) (if (null l) nil (app (rev (cdr l)) (cons (car l) nil))))
(de app (x y) (if (null x) y (cons (car x) (app (cdr x) y))))
```

Nous avons maintenant :

$$(\text{rev} (\text{app} (\text{rev } A) (\text{cons } a \text{ nil}))) = (\text{cons } a (\text{rev} (\text{rev } A)))$$

Et sommes toujours bloqués. Donc : lemme.

Mais, **généralisons** d'abord : $(\text{rev } A) \mapsto C$, ce qui donne le lemme :

$$(\text{rev} (\text{app } C (\text{cons } a \text{ nil}))) = (\text{cons } a (\text{rev } C))$$

Induction sur C :

$$\text{La base : } (\text{rev} (\text{app } \text{nil} (\text{cons } a \text{ nil}))) = (\text{cons } a (\text{rev } \text{nil}))$$

$$(\text{rev} (\text{cons } a \text{ nil})) = (\text{cons } a \text{ nil})$$

$$(\text{app} (\text{rev } \text{nil}) (\text{cons } a \text{ nil})) = (\text{cons } a \text{ nil})$$

$$(\text{app } \text{nil} (\text{cons } a \text{ nil})) = (\text{cons } a \text{ nil})$$

$$(\text{cons } a \text{ nil}) = (\text{cons } a \text{ nil})$$



$$(\text{rev} (\text{rev } A)) = A \quad (3)$$

```
(de rev (l) (if (null l) nil (app (rev (cdr l)) (cons (car l) nil))))  
(de app (x y) (if (null x) y (cons (car x) (app (cdr x) y))))
```

Le pas : $(\text{rev} (\text{app} (\text{cons } c \text{ C})(\text{cons } a \text{ nil}))) = (\text{cons } a (\text{rev} (\text{cons } c \text{ C})))$

$(\text{rev} (\text{cons } c (\text{app } C (\text{cons } a \text{ nil})))) = (\text{cons } a (\text{app} (\text{rev } C) (\text{cons } c \text{ nil})))$

$(\text{app} (\text{rev} (\text{app } C (\text{cons } a \text{ nil}))) (\text{cons } c \text{ nil})) = (\text{cons } a (\text{app} (\text{rev } C) (\text{cons } c \text{ nil})))$

Fertilisation avec $(\text{rev} (\text{app } C (\text{cons } a \text{ nil}))) = (\text{cons } a (\text{rev } C))$:

$(\text{app} (\text{cons } a (\text{rev } C)) (\text{cons } c \text{ nil})) = (\text{cons } a (\text{app} (\text{rev } C) (\text{cons } c \text{ nil})))$

$(\text{cons } a (\text{app} (\text{rev } C) (\text{cons } c \text{ nil}))) = (\text{cons } a (\text{app} (\text{rev } C) (\text{cons } c \text{ nil})))$ ✓

Et voilà, la démonstration de $(\text{rev} (\text{rev } A)) = A$ est faite !

Quelques axiomes de plus

Si nous ajoutons à notre liste d'axiomes quelques-uns décrivant les opérations arithmétiques, tel que :

$$(1 + (1 - n)) = n$$

$$(1 - (1 + n)) = n$$

nous pouvons également démontrer des programmes contenant des expressions arithmétiques

Commutativité de l'addition (1)

Soit le programme :

(de p (m n)
(if (equal m 0) n (1+ (p (1- m) n))))

et démontrons :

$\forall x, y \in \text{ nombres naturelles } (p \ x \ y) = (p \ y \ x)$

Commutativité de l'addition (2)

(de p (m n) (if (equal m 0) n (1+ (p (1- m) n))))

1. Choix de la variables d'induction : m
2. Preuve de la base : m = 0

$$(p\ 0\ y) = (p\ y\ 0)$$

$$(if\ (equal\ 0\ 0)\ y\ (1+ (p\ (1- 0)\ y))) = (p\ y\ 0)$$

$$y = (p\ y\ 0)$$

Blocké !

Commutativité de l'addition (3)

(de p (m n) (if (equal m 0) n (1+ (p (1- m) n))))

Donc lemme : $y = (p y 0)$

induction sur n

base : $0 = (p 0 0)$

$$0 = 0$$

pas : $(1+ y) = (p (1+ y) 0)$

$$(1+ y) = (\text{if } (\text{equal } (1+ y) 0) 0 (1+ (p (1- (1+ y)) 0)))$$

$$(1+ y) = (1+ (p (1- (1+ y)) 0))$$

$$(1+ y) = (1+ (p y 0))$$

et par cross-fertilisation sur : $y = (p y 0)$

$$(1+ y) = (1+ y)$$

✓ pour la base

Commutativité de l'addition (4)

(de p (m n) (if (equal m 0) n (1+ (p (1- m) n))))

3. Preuve du pas :

$$(p (1+ x) y) = (p y (1+ x))$$

$$(if (= (1+ x) 0) y (1+ (p (1- (1+ x)) y))) = (p y (1+ x))$$

$$(1+ (p (1- (1+ x)) y)) = (p y (1+ x))$$

$$(1+ (p x y)) = (p y (1+ x))$$

Cross-fertilisation avec $(p x y) = (p y x)$ donne :

$$(1+ (p y x)) = (p y (1+ x))$$

Blocké ! Donc lemme !

Commutativité de l'addition (5)

(de p (m n) (if (equal m 0) n (1+ (p (1- m) n))))

Preuve du lemme $(1+ (p y x)) = (p y (1+ x))$:

base : $(1+ (p 0 x)) = (p 0 (1+ x))$

$$(1+ x) = (1+ x)$$

Pas : $(1+ (p (1+ y) x)) = (p (1+ y)(1+ x))$

$$(1+ (1+ (p (1- (1+ y)) x))) = (1+ (p (1- (1+ y))(1+ x)))$$

$$(1+ (1+ (p y x))) = (1+ (p y (1+ x)))$$

Cross-fertilisation sur $(1+ (p y x)) = (p y (1+ x))$ donne :

$$(1+ (p y (1+ x))) = (1+ (p y (1+ x))) \quad \checkmark$$

Exercices

Avec les fonctions **app** et **length** ci dessous :

```
(de app (x y)
  (if (null x) y (cons (car x) (app (cdr x) y))))
```

```
(de length (l)
  (if (null l) 0 (1+ (length (cdr l)))))
```

Démontrez :

$$\forall A, B \in \text{listes} \text{ (length (app A B)) = (length (app B A))}$$

Et en utilisant la fonction **p** des pages précédentes, démontrez :

$$\forall A, B \in \text{listes} \text{ (length (app A B)) = (p (length A)(length B))}$$