funoptimizer: optimisation de programmes fonctionnels

https://gaufre.informatique.univ-paris-diderot.fr/runser/runser-zhang-plong-2022

Maryline Zhang et Laure Runser

Université de Paris

2023

Objectif initial et problématique

Comment augmenter l'efficacité d'un programme compilé pour un langage fonctionnel ?

- Transformations de programme du langage Core
- Constant folding
- Preuve de correction
- Aspect pédagogique

Ressource principale: thèse de M. Santos
Compilation by Transformation in Non-Strict Functional Languages

Choix de développement

- Pas d'interface utilisateur
- Preuve de l'algorithme au lieu de faire d'autres transformations
- Partie réduite du langage Core

Expression	*Expr*	→	*Expr Atom*	Application
		|	*Expr ty*	Type application
		|	$\lambda\ var_1 \ldots var_n$ -> *Expr*	Lambda abstraction
		|	Λ *ty* -> *Expr*	Type abstraction
		|	`case` *Expr* `of` *Alts*	Case expression
		|	`let` *Binding* `in` *Expr*	Local definition
		|	*con* $Atom_1 \ldots Atom_n$	Constructor $n \geq 0$
		|	*prim* $Atom_1 \ldots Atom_n$	Primitive $n \geq 0$
		|	*Atom*	

Syntaxe de Core

$$
\begin{array}{lll}
t, u, v & ::= & \\
& \mid \quad b & \text{(base)} \\
& \mid \quad \textbf{fun}\ (x : T) = t & \text{(abstraction)} \\
& \mid \quad t\ b & \text{(function application)} \\
& \mid \quad \textbf{let}\ x = t\ \textbf{in}\ u & \text{(let binding)} \\
& \mid \quad \textbf{if}\ t\ \textbf{then}\ u\ \textbf{else}\ v & \text{(conditional)} \\
& \mid \quad \textbf{fun}\ \langle X \rangle = t & \text{(type abstraction)} \\
& \mid \quad t\ \langle T \rangle & \text{(type application)} \\
& \mid \quad (t : T) & \text{(type annotation)}
\end{array}
$$

$$
\begin{array}{lll}
b & ::= & \\
& \mid \quad x & \text{(variable)} \\
& \mid \quad \textbf{true} & \text{(true)} \\
& \mid \quad \textbf{false} & \text{(false)}
\end{array}
$$

$$
\begin{array}{lll}
T, S & ::= & \\
& \mid \quad X & \text{(type variable)} \\
& \mid \quad \textbf{bool} & \text{(bool type)} \\
& \mid \quad S \to T & \text{(function type)} \\
& \mid \quad \forall X.T & \text{(polymorphic type)}
\end{array}
$$

Constant folding

Règles

$$\frac{}{(\textbf{fun}\ (x : T) = t)\ b \rightsquigarrow t\{x \mapsto b\}} \qquad \frac{}{\textbf{if true then}\ t\ \textbf{else}\ u \rightsquigarrow t}$$

$$\frac{}{(\textbf{fun}\ \langle X \rangle = t)\ \langle T \rangle \rightsquigarrow t\{X \mapsto T\}} \qquad \frac{}{\textbf{if false then}\ t\ \textbf{else}\ u \rightsquigarrow u}$$

Exemple

$$\begin{aligned}
&\textbf{if}\ ((\textbf{fun}\ (x : \textbf{bool}) = x)\ \textbf{false})\ \textbf{then}\ x\ \textbf{else}\ y \\
&\rightsquigarrow \textbf{if false then}\ x\ \textbf{else}\ y \\
&\rightsquigarrow y
\end{aligned}$$

Vue d'ensemble

Structure du dépôt

```
- lib/
  |- atom
  |- terms
  |- types
  |- stack
- test/
- docs/
  |- syntax/
```

Etapes de réalisation

- Mise en place de la syntaxe, de l'alpha-renommage, et du typechecker (PR !1, !5, !6)
- Ajout des contextes d'évaluation (PR !4)
- Ajout de la simplification (PR !9)
- Preuve de correction (PR !10)
- Rectification du code grâce à la preuve (PR !11)

Compétences techniques

- Construire une preuve en autonomie
- Programmation en OCaml
- Ecriture de LaTex avec des macros
- Utilisation professionnelle de git: code review, git flow

Répartition du travail équitable

Principales difficultés

- Découverte de nouveaux outils en OCaml et LaTeX
- Utilisation raisonnée de git
- Nouveautés théoriques
- Schéma de preuve

Algorithme: go

$t = \textbf{if}\ (\textbf{fun}\ (x : \textbf{bool}) = x)\ \textbf{true}\ \textbf{then}\ \textbf{fun}\ \langle X \rangle = u\ \textbf{else}\ \textbf{fun}\ \langle X \rangle = v$

$\text{go}\ ((\textbf{fun}\ \langle Y \rangle = t\ \langle Y \rangle)\ \langle \textbf{bool} \rangle)^{\text{id}}\ \varepsilon$

Algorithme: go

$$t = \textbf{if } (\textbf{fun } (x : \textbf{bool}) = x) \textbf{ true then fun } \langle X \rangle = u \textbf{ else fun } \langle X \rangle = v$$

$$\text{go } ((\textbf{fun } \langle Y \rangle = t \langle Y \rangle) \langle \textbf{bool} \rangle)^{\text{id}} \varepsilon$$

$$= \text{go } (\textbf{fun } \langle Y \rangle = t \langle Y \rangle)^{\text{id}} (\square^{\text{id}} \textbf{ bool})$$

Algorithme: go

$$t = \textbf{if } (\textbf{fun } (x : \textbf{bool}) = x) \textbf{ true then fun } \langle X \rangle = u \textbf{ else fun } \langle X \rangle = v$$

$$\begin{aligned}
&\text{go } ((\textbf{fun } \langle Y \rangle = t \langle Y \rangle) \langle \textbf{bool} \rangle)^{\text{id}} \varepsilon \\
&= \text{go } (\textbf{fun } \langle Y \rangle = t \langle Y \rangle)^{\text{id}} (\square^{\text{id}} \textbf{ bool}) \\
&= \text{go } (t \langle Y \rangle)^{Y \mapsto \textbf{bool}} \varepsilon
\end{aligned}$$

Algorithme: go

$$t = \textbf{if } (\textbf{fun } (x : \textbf{bool}) = x) \textbf{ true then fun } \langle X \rangle = u \textbf{ else fun } \langle X \rangle = v$$

$$\begin{aligned}
&\text{go } ((\textbf{fun } \langle Y \rangle = t \langle Y \rangle) \langle \textbf{bool} \rangle)^{\text{id}} \varepsilon \\
&= \text{go } (\textbf{fun } \langle Y \rangle = t \langle Y \rangle)^{\text{id}} (\square^{\text{id}} \textbf{ bool}) \\
&= \text{go } (t \langle Y \rangle)^{Y \mapsto \textbf{bool}} \varepsilon \\
&= \text{go } t^{Y \mapsto \textbf{bool}} (\square^{Y \mapsto \textbf{bool}} Y)
\end{aligned}$$

Algorithme: go

$$t = \textbf{if } (\textbf{fun } (x : \textbf{bool}) = x) \textbf{ true then fun } \langle X \rangle = u \textbf{ else fun } \langle X \rangle = v$$

$$\begin{aligned}
&\text{go } ((\textbf{fun } \langle Y \rangle = t \langle Y \rangle) \langle \textbf{bool} \rangle)^{\text{id}} \varepsilon \\
&= \text{go } (\textbf{fun } \langle Y \rangle = t \langle Y \rangle)^{\text{id}} (\square^{\text{id}} \textbf{ bool}) \\
&= \text{go } (t \langle Y \rangle)^{Y \mapsto \textbf{bool}} \varepsilon \\
&= \text{go } t^{Y \mapsto \textbf{bool}} (\square^{Y \mapsto \textbf{bool}} Y) \\
&= \text{go } ((\textbf{fun } (x : \textbf{bool}) = x) \textbf{ true})^{Y \mapsto \textbf{bool}} s
\end{aligned}$$

avec $s = \textbf{if } \square^{Y \mapsto \textbf{bool}} \textbf{ then fun } \langle X \rangle = u \textbf{ else fun } \langle X \rangle = v \triangleright \square^{Y \mapsto \textbf{bool}} Y$

Algorithme: go

$t = \textbf{if } (\textbf{fun } (x : \textbf{bool}) = x) \textbf{ true then fun } \langle X \rangle = u \textbf{ else fun } \langle X \rangle = v$

$\text{go } ((\textbf{fun } \langle Y \rangle = t \langle Y \rangle) \langle \textbf{bool} \rangle)^{\text{id}} \varepsilon$

$= \text{go } (\textbf{fun } \langle Y \rangle = t \langle Y \rangle)^{\text{id}} (\square^{\text{id}} \textbf{ bool})$

$= \text{go } (t \langle Y \rangle)^{Y \mapsto \textbf{bool}} \varepsilon$

$= \text{go } t^{Y \mapsto \textbf{bool}} (\square^{Y \mapsto \textbf{bool}} Y)$

$= \text{go } ((\textbf{fun } (x : \textbf{bool}) = x) \textbf{ true})^{Y \mapsto \textbf{bool}} s$

avec $s = \textbf{if } \square^{Y \mapsto \textbf{bool}} \textbf{ then fun } \langle X \rangle = u \textbf{ else fun } \langle X \rangle = v \triangleright \square^{Y \mapsto \textbf{bool}} Y$

$= \text{go } (\textbf{fun } (x : \textbf{bool}) = x)^{Y \mapsto \textbf{bool}} (\square^{Y \mapsto \textbf{bool}} \textbf{ true} \triangleright s)$

Algorithme: go

$$t = \textbf{if } (\textbf{fun } (x : \textbf{bool}) = x)\ \textbf{true } \textbf{then } \textbf{fun } \langle X \rangle = u \textbf{ else } \textbf{fun } \langle X \rangle = v$$

$$\text{go } ((\textbf{fun } \langle Y \rangle = t\ \langle Y \rangle)\ \langle \textbf{bool} \rangle)^{\text{id}}\ \varepsilon$$

$$= \text{go } (\textbf{fun } \langle Y \rangle = t\ \langle Y \rangle)^{\text{id}}\ (\square^{\text{id}}\ \textbf{bool})$$

$$= \text{go } (t\ \langle Y \rangle)^{Y \mapsto \textbf{bool}}\ \varepsilon$$

$$= \text{go } t^{Y \mapsto \textbf{bool}}\ (\square^{Y \mapsto \textbf{bool}}\ Y)$$

$$= \text{go } ((\textbf{fun } (x : \textbf{bool}) = x)\ \textbf{true})^{Y \mapsto \textbf{bool}}\ s$$

$$\text{avec } s = \textbf{if } \square^{Y \mapsto \textbf{bool}} \textbf{ then } \textbf{fun } \langle X \rangle = u \textbf{ else } \textbf{fun } \langle X \rangle = v \triangleright \square^{Y \mapsto \textbf{bool}}\ Y$$

$$= \text{go } (\textbf{fun } (x : \textbf{bool}) = x)^{Y \mapsto \textbf{bool}}\ (\square^{Y \mapsto \textbf{bool}}\ \textbf{true} \triangleright s)$$

$$= \text{go } x^{Y \mapsto \textbf{bool}, x \mapsto \textbf{true}}\ s$$

Algorithme: go

$$t = \textbf{if } (\textbf{fun } (x : \textbf{bool}) = x)\ \textbf{true } \textbf{then } \textbf{fun } \langle X \rangle = u \textbf{ else } \textbf{fun } \langle X \rangle = v$$

$$\text{go } ((\textbf{fun } \langle Y \rangle = t\ \langle Y \rangle)\ \langle \textbf{bool} \rangle)^{\text{id}}\ \varepsilon$$

$$= \text{go } (\textbf{fun } \langle Y \rangle = t\ \langle Y \rangle)^{\text{id}}\ (\square^{\text{id}}\ \textbf{bool})$$

$$= \text{go } (t\ \langle Y \rangle)^{Y \mapsto \textbf{bool}}\ \varepsilon$$

$$= \text{go } t^{Y \mapsto \textbf{bool}}\ (\square^{Y \mapsto \textbf{bool}}\ Y)$$

$$= \text{go } ((\textbf{fun } (x : \textbf{bool}) = x)\ \textbf{true})^{Y \mapsto \textbf{bool}}\ s$$

$$\text{avec } s = \textbf{if } \square^{Y \mapsto \textbf{bool}} \textbf{ then } \textbf{fun } \langle X \rangle = u \textbf{ else } \textbf{fun } \langle X \rangle = v \triangleright \square^{Y \mapsto \textbf{bool}}\ Y$$

$$= \text{go } (\textbf{fun } (x : \textbf{bool}) = x)^{Y \mapsto \textbf{bool}}\ (\square^{Y \mapsto \textbf{bool}}\ \textbf{true} \triangleright s)$$

$$= \text{go } x^{Y \mapsto \textbf{bool}, x \mapsto \textbf{true}}\ s$$

$$= \text{go } (\textbf{fun } \langle X \rangle = u)^{Y \mapsto \textbf{bool}}\ (\square^{Y \mapsto \textbf{bool}}\ Y)$$

Algorithme: go

$$t = \textbf{if } (\textbf{fun } (x : \textbf{bool}) = x)\ \textbf{true } \textbf{then } \textbf{fun } \langle X \rangle = u \textbf{ else } \textbf{fun } \langle X \rangle = v$$

$$\text{go } ((\textbf{fun } \langle Y \rangle = t\ \langle Y \rangle)\ \langle \textbf{bool} \rangle)^{\text{id}}\ \varepsilon$$

$$= \text{go } (\textbf{fun } \langle Y \rangle = t\ \langle Y \rangle)^{\text{id}}\ (\square^{\text{id}}\ \textbf{bool})$$

$$= \text{go } (t\ \langle Y \rangle)^{Y \mapsto \textbf{bool}}\ \varepsilon$$

$$= \text{go } t^{Y \mapsto \textbf{bool}}\ (\square^{Y \mapsto \textbf{bool}}\ Y)$$

$$= \text{go } ((\textbf{fun } (x : \textbf{bool}) = x)\ \textbf{true})^{Y \mapsto \textbf{bool}}\ s$$

$$\text{avec } s = \textbf{if } \square^{Y \mapsto \textbf{bool}} \textbf{ then } \textbf{fun } \langle X \rangle = u \textbf{ else } \textbf{fun } \langle X \rangle = v \triangleright \square^{Y \mapsto \textbf{bool}}\ Y$$

$$= \text{go } (\textbf{fun } (x : \textbf{bool}) = x)^{Y \mapsto \textbf{bool}}\ (\square^{Y \mapsto \textbf{bool}}\ \textbf{true} \triangleright s)$$

$$= \text{go } x^{Y \mapsto \textbf{bool}, x \mapsto \textbf{true}}\ s$$

$$= \text{go } (\textbf{fun } \langle X \rangle = u)^{Y \mapsto \textbf{bool}}\ (\square^{Y \mapsto \textbf{bool}}\ Y)$$

$$= \text{go } u^{Y \mapsto \textbf{bool}, X \mapsto \textbf{bool}}\ \varepsilon$$

go: spécification complète

$$\mathsf{go}\ b^{\rho}\ s = \begin{cases} \mathsf{go}\ u^{\rho'}\ s' & \text{if } b\,\{\rho\} = \mathbf{true} \text{ and } s = \mathbf{if}\ \square^{\rho'}\ \mathbf{then}\ u\ \mathbf{else}\ v \triangleright s' \\ \mathsf{go}\ v^{\rho'}\ s' & \text{if } b\,\{\rho\} = \mathbf{false} \text{ and } s = \mathbf{if}\ \square^{\rho'}\ \mathbf{then}\ u\ \mathbf{else}\ v \triangleright s' \\ \mathsf{plug}\ s\ (b\,\{\rho\}) & \text{otherwise} \end{cases}$$

$$\mathsf{go}\ (\mathbf{fun}\ (x:T) = t)^{\rho}\ s = \begin{cases} \mathsf{go}\ t^{\rho, x \mapsto b\{\rho'\}}\ s' & \text{if } s = \square^{\rho'}\ b \triangleright s' \\ \mathsf{plug}\ s\ (\mathbf{fun}\ (y:T\,\{\rho\}) = (\mathsf{go}\ t^{\rho, x \mapsto y}\ \varepsilon)) & \text{otherwise} \\ & \text{with } y \notin FV(t\,\{\rho\}) \end{cases}$$

$$\mathsf{go}\ (\mathbf{fun}\ \langle X \rangle = t)^{\rho}\ s = \begin{cases} \mathsf{go}\ t^{\rho, X \mapsto S\{\rho'\}}\ s' & \text{if } s = \square^{\rho'}\ S \triangleright s' \\ \mathsf{plug}\ s\ (\mathbf{fun}\ \langle Y \rangle = (\mathsf{go}\ t^{\rho, X \mapsto Y}\ \varepsilon)) & \text{otherwise} \\ & \text{with } Y \notin FV(t\,\{\rho\}) \end{cases}$$

$$\mathsf{go}\ (t\ b)^{\rho}\ s = \mathsf{go}\ t^{\rho}\ (\square^{\rho}\ b \triangleright s)$$

$$\mathsf{go}\ (t\ \langle S \rangle)^{\rho}\ s = \mathsf{go}\ t^{\rho}\ (\square^{\rho}\ S \triangleright s)$$

$$\mathsf{go}\ (\mathbf{if}\ t\ \mathbf{then}\ u\ \mathbf{else}\ v)^{\rho}\ s = \mathsf{go}\ t^{\rho}\ (\mathbf{if}\ \square^{\rho}\ \mathbf{then}\ u\ \mathbf{else}\ v \triangleright s)$$

$$\mathsf{go}\ (\mathbf{let}\ x = t\ \mathbf{in}\ u)^{\rho}\ s = \mathsf{plug}\ s\ t' \text{ with } t' = (\mathbf{let}\ y = (\mathsf{go}\ t^{\rho}\ \varepsilon)\ \mathbf{in}\ (\mathsf{go}\ u^{\rho, x \mapsto y}\ \varepsilon)) \text{ and } y \notin FV(v\,\{\rho\})$$

$$\mathsf{go}\ (t:T)^{\rho}\ s = \mathsf{go}\ t^{\rho}\ s$$

$$\mathsf{plug}\ \varepsilon\ t = t$$

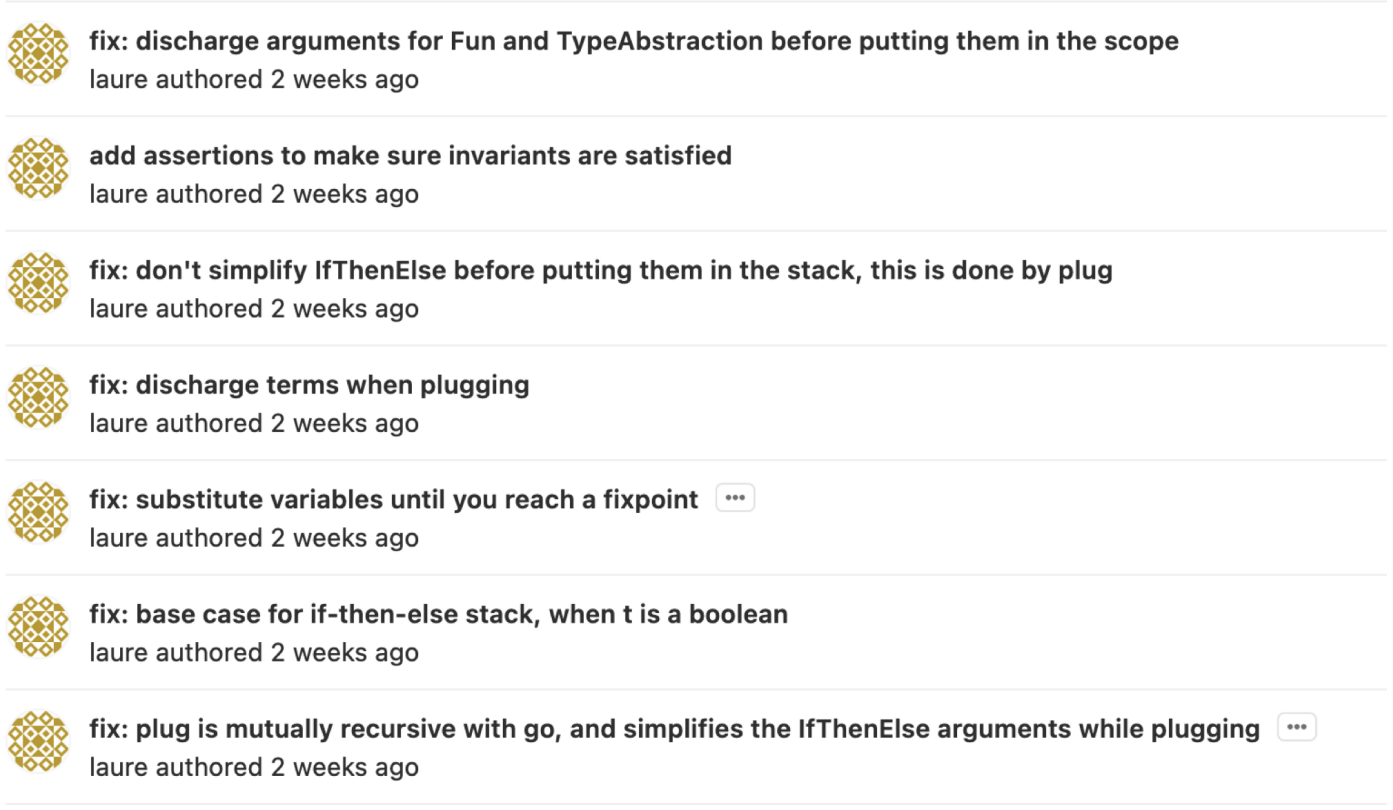
$$\mathsf{plug}\ (\square^{\rho}\ b \triangleright s)\ t = \mathsf{plug}\ s\ (t\ (b\,\{\rho\}))$$

$$\mathsf{plug}\ (\square^{\rho}\ S \triangleright s)\ t = \mathsf{plug}\ s\ (t\ \langle S\,\{\rho\} \rangle)$$

$$\mathsf{plug}\ (\mathbf{if}\ \square^{\rho}\ \mathbf{then}\ u\ \mathbf{else}\ v \triangleright s)\ t = \mathsf{plug}\ s\ (\mathbf{if}\ t\ \mathbf{then}\ \mathsf{go}\ u^{\rho}\ \varepsilon\ \mathbf{else}\ \mathsf{go}\ v^{\rho}\ \varepsilon)$$

Testabilité

- 142 tests, qui s'exécutent en 0.019s
- La preuve a permis de corriger le code



Conclusion

On a appris:

- à bien programmer en OCaml
- à construire une preuve en autonomie
- à écrire un document conséquent en LaTeX
- à adapter nos objectifs

Une version 2.0:

- avec une interface et un parser
- avec du let-floating et de l'inlining
- avec un outil de comparaison de performance

A changer:

- commencer après le cours de semantique
- problème d'adéquation entre le problème et les objets qu'on a l'habitude de manipuler (preuve de programme)
- être plus strictes avec git pour éviter des gros conflits