

Optimisation de la vitesse de compilation par la fusion entre inférence de types et analyse syntaxique

Enogad Le Biavant–Frederic

Alain René Lesage MPI

2025

L'idée

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntaxique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing
Théorie des types

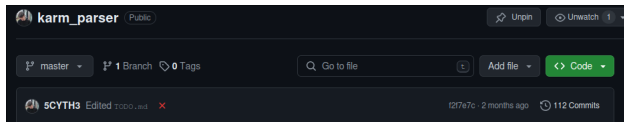
Résultats

Formalisation

Conclusion

Annexe

Karm, 2022



Comment optimiser la vitesse de compilation en fusionnant
analyse syntaxique et sémantique ?

L'idée

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntactique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing
Théorie des types

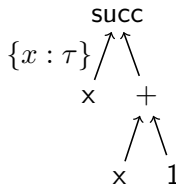
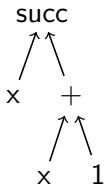
Résultats

Formalisation

Conclusion

Annexe

$\Gamma = \{+ : int \rightarrow int \rightarrow int\}$
`let succ = $\lambda x. (+ x 1)$`



L'idée

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntaxique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing
Théorie des types

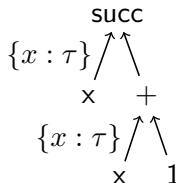
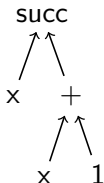
Résultats

Formalisation

Conclusion

Annexe

$\Gamma = \{+ : int \rightarrow int \rightarrow int, x : \tau\}$
`let succ = $\lambda x. (+ x 1)$`



L'idée

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntactique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing
Théorie des types

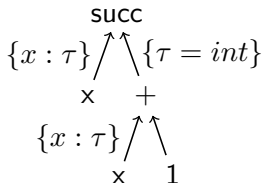
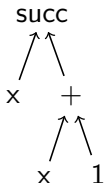
Résultats

Formalisation

Conclusion

Annexe

$\Gamma = \{+ : int \rightarrow int \rightarrow int, x : \tau\}$
`let succ = $\lambda x. (+ x 1)$`



L'idée

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntactique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing
Théorie des types

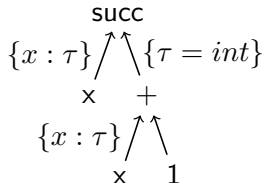
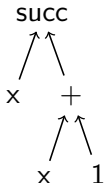
Résultats

Formalisation

Conclusion

Annexe

$\Gamma = \{+ : int \rightarrow int \rightarrow int, x : \tau\}$
 $\text{let } succ = \lambda x. (+ x 1)$



Parsing récursif descendant

Grammaire

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntactique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing

Théorie des types

Résultats

Formalisation

Conclusion

Annexe

<i>program</i>	::=	<i>expr</i>
<i>expr</i>	::=	<i>abs</i>
		<i>app</i>
		<i>letbinding</i>
<i>app</i>	::=	<i>term</i> [{ <i>term</i> }]
<i>abs</i>	::=	"\" <i>id</i> \".\" <i>expr</i>
<i>letbinding</i>	::=	"let\" <i>id</i> \"=\" <i>expr</i> \"in\" <i>expr</i>
<i>term</i>	::=	string
		int
		bool
		<i>id</i>
		"(\" <i>expr</i> \")"

TT - Définitions

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntaxique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing

Théorie des types

Résultats

Formalisation

Conclusion

Annexe

Définition de Type : classification de termes (Church).
Théorie de travail : Lambda calcul simplement typé (LCST),
polymorphique.

$$\text{let } id = \lambda x.x : \forall \sigma \rightarrow \sigma$$

Hindley-Milner

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntactique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing

Théorie des types

Résultats

Formalisation

Conclusion

Annexe

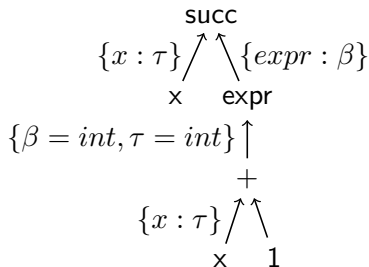
$$\frac{x : \sigma \in \Gamma}{\Gamma \vdash x : \sigma} \text{ var}$$

$$\frac{\Gamma, x : \tau \vdash e : \tau'}{\Gamma \vdash \lambda x. e : \tau \rightarrow \tau'} \text{ abs}$$

$$\frac{\Gamma \vdash f : \tau \rightarrow \tau' \quad \Gamma \vdash e : \tau}{\Gamma \vdash f e : \tau'} \text{ app}$$

Algorithme W

- 1 Assignation de variables de types aux expressions
- 2 Génération de contraintes
- 3 Substitutions
- 4 Unification
- 5 Instantiation, généralisation



Résultats

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntaxique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing
Théorie des types

Résultats

Formalisation

Conclusion

Annexe

$$\begin{aligned}\mathcal{W} &: \tilde{\Gamma} \times Expr \rightarrow Subst \times Type \\ \mathcal{W}^* &: \tilde{\Gamma} \times L \rightarrow Subst \times \Gamma \times Expr \times L\end{aligned}$$

Machine : i7 5th gen 3.00Ghz

Fichier de test : 1000 premiers nombres de church

Version non optimisée : $\approx 26.37s$

Version optimisée : $\approx 3.88s$

Fichier de test : 10K application de fonction successeur

Version non optimisée : $\approx 1.48s$

Version optimisée : $\approx 1.51s$

Formalisation

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntaxique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing
Théorie des types

Résultats

Formalisation

Conclusion

Annexe

Automates d'arbres : $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$

- \mathcal{F} : Alphabet gradué (fonction d'arité ar)

On peut décrire la grammaire comme une NRTG

Conclusion

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntaxique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing

Théorie des types

Résultats

Formalisation

Conclusion

Annexe

Bénéfices

Durée inférieure lorsque beaucoup d'unifications

Plus permissif envers les grammaires très imbriquées que
les systèmes traditionnels

Annexe

Optimisation
de la vitesse
de compilation
par la fusion
entre inférence
de types et
analyse
syntaxique

Enogad Le
Biavant-
Frederic

Présentation
générale

Définitions

Parsing

Théorie des types

Résultats

Formalisation

Conclusion

Annexe