Invariance in the Lambda Calculus through Explicit Substitutions

Haileselassie Gaspar

Vrije Universiteit Amsterdam

11-07-2025

Invariance in the Lambda Calculus through Explicit Substitutions
Haileselassie Gaspar
Vrije Universiteit Amsterdam
11-07-2025

Contents

Contents
Introduction and Background
The Size-Explosion Problem
Linear Substitution Calculus
High-Level Implementation Systems
Examples
Conclusion

Equivalence and Invariance of Models

- The Church-Turing Thesis - Turing, Kleene, Church, Rosser
- The Invariance Thesis - Van Embde Boas

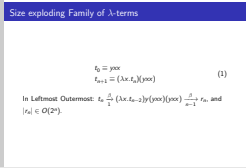
Equivalence and Invariance of Models
- The Church-Turing Thesis - Turing, Kleene, Church, Rosser
- The Invariance Thesis - Van Embde Boas

Turing machines are the foundational measure of computational complexity so when we talk about equivalence and invariance we refer to it WITH RESPECT TO TURING MACHINES. We will study INVARIANCE through a cost model. We will only talk about time invariance in this presentation.

Size exploding Family of λ-terms

$$t_0 \equiv yxx$$
$$t_{n+1} \equiv (\lambda x.t_n)(yxx) \quad (1)$$

In Leftmost Outermost: $t_n \xrightarrow[1]{\beta} (\lambda x.t_{n-2})y(yxx)(yxx) \xrightarrow[n-1]{\beta} r_n$, and $|r_n| \in O(2^n)$.



When we talk about the size explosion we refer to it in terms of how a Turing machine would represent this term, which obviously takes 2^n steps since space complexity is a lower bound for time complexity on Turing machines

Syntax and operational semantics

- Syntax

$$
\begin{aligned}
t, u &::= \ x \mid \lambda_{lsc} x.t \mid tu \mid t[x \leftarrow u] \\
S &::= \ \langle \cdot \rangle \mid \lambda_{lsc} x.S \mid St \mid tS \mid S[x \leftarrow t] \\
L &::= \ \langle \cdot \rangle \mid L[x \leftarrow t]
\end{aligned} \tag{2}
$$

- Operational Semantics

$$
\begin{aligned}
L\langle \lambda_{lsc} x.t \rangle u &\rightarrow_{dB} L\langle t[x \leftarrow u] \rangle \\
S\langle x \rangle [x \leftarrow u] &\rightarrow_{ls} S\langle u \rangle [x \leftarrow u]
\end{aligned} \tag{3}
$$

Syntax and operational semantics
Syntax
$t, u ::= x \mid \lambda_{lsc} x.t \mid tu \mid t[x \leftarrow u]$
$S ::= \langle \cdot \rangle \mid \lambda_{lsc} x.S \mid St \mid tS \mid S[x \leftarrow t]$ (2)
$L ::= \langle \cdot \rangle \mid L[x \leftarrow t]$
Operational Semantics
$L\langle \lambda_{lsc} x.t \rangle u \rightarrow_{dB} L\langle t[x \leftarrow u] \rangle$
$S\langle x \rangle [x \leftarrow u] \rightarrow_{ls} S\langle u \rangle [x \leftarrow u]$ (3)

Unfolding of Shared terms

We introduce the operation $\downarrow$ in order to convert λ_{LSC}-terms to regular λ-terms.

$$t[x \leftarrow u]\downarrow = t\downarrow\{x \leftarrow u\downarrow\} \tag{4}$$

And the contextual unfolding of a term:

$$t\downarrow_{S[x\rightarrow u]} = t\downarrow_S\{x \rightarrow u\downarrow\} \tag{5}$$

Unfolding of Shared terms
We introduce the operation ↓ in order to convert λ_{LSC}-terms to regular λ-terms.
$t[x \leftarrow u]\downarrow = t\downarrow\{x \leftarrow u\downarrow\}$ (4)
And the contextual unfolding of a term:
$t\downarrow_{S[x\rightarrow u]} = t\downarrow_S\{x \rightarrow u\downarrow\}$ (5)

Given a pair $(\rightsquigarrow, \rightsquigarrow_X)$ and a derivation $\rho : t \rightsquigarrow_X u$:

1. Normal Form Equality
2. Projection .
3. Trace
4. Syntactic Bound

This leads to:

1. Normalization
2. Quadratic Bound

Given a pair $(\rightsquigarrow, \rightsquigarrow_X)$ and a derivation $\rho : t \rightsquigarrow_X u$:
1. Normal Form Equality
2. Projection .
3. Trace
4. Syntactic Bound
This leads to:
1. Normalization
2. Quadratic Bound

If a term is in LSC normalform it is on lambda normal form
For any derivation in the LSC, the unfolding of the derivation leads to the unfolded final term, and the number of dB steps is the same as the size of the unfolded derivation
The number of explicit substitutions in the final term is exactly the amount of dB steps in the reduction
the length of substitution steps from u is bounded by the number of explicit substitutions

Useful Derivations

Applicative context: $A = S\langle Lt \rangle$.
A useful step is either a dB-step or a ls-step $S\langle x \rangle \to S\langle r \rangle$ so that the unfolding $r \downarrow_S$:

1. Either contains a β-redex
2. Or is an abstraction and S is an applicative context.

Useful Derivations
Applicative context: $A = S\langle Lt \rangle$.
A useful step is either a dB-step or a ls-step $S\langle x \rangle \to S\langle r \rangle$ so that the unfolding $r \downarrow_S$:
1. Either contains a β-redex
2. Or is an abstraction and S is an applicative context.

Size exploding Terms revisited

Taking $u = yxx$ for readability:

$$t_2 \equiv (\lambda x.(\lambda x.(yxx))(yxx))(yxx) \xrightarrow[2]{\beta} y(yuu)(yuu) \equiv r_2$$
$$t_2 \xrightarrow[2]{dB} (yxx)[x \leftarrow yxx][x \leftarrow yxx] \equiv r_2' \qquad (6)$$

Size exploding Terms revisited
Taking $u = yxx$ for readability:
$t_2 \equiv (\lambda x.(\lambda x.(yxx))(yxx))(yxx) \xrightarrow[2]{\beta} y(yuu)(yuu) \equiv r_2$
$t_2 \xrightarrow[2]{dB} (yxx)[x \leftarrow yxx][x \leftarrow yxx] \equiv r_2'$ (6)

Conclusion

1. The representation of size-exploding terms is done in time polynomial to the size of the initial term.
2. The size-explosion problem is solved.

Conclusion
The representation of size-exploding terms is done in time polynomial to the size of the initial term.
The size-explosion problem is solved.