Rewriting, Explicit Substitutions and Normalisation

XXXVI Escola de Verão do MAT
Universidade de Brasilia

Part 3/3

Eduardo Bonelli

LIFIA (Fac. de Informática, UNLP, Arg.) and CONICET
eduardo@lifia.info.unlp.edu.ar

February, 2006

Structure of Today's Talk

1. Explicit Substitutions
2. Normalisation for Calculi with ES
3. Additional Problems in Explicit Substitutions

Substitution: A critique

$$M\{x/N\}$$

- Complex
  - Requires traversing the structure of M
  - Requires renaming of bound variables in order to avoid variable capture
  - May create multiple copies of N
  - A number of proof assistants had buggy implementations of substitution. Eg. LCF

"Every year or so, LCF users found some serious bug. (Most were due to bound variable clashes in strange situations.)" *Isabelle: The Next 700 Theorem Provers*, L . Paulson, P. Odifreddi (editor), Logic and Computer Science (Academic Press, 1990), 361-386.

Substitution: A critique

- Treat substitution seriously
  - Replace it in favour of a more "atomic" encoding
  - Haul this encoding into your object language
  - Consider your encoding to be as "structure preserving" as possible (this rules out Combinatory Logic)
- Benefits
  - Fine-grained control of substitution propagation
  - Richer dynamics (eg. choose to avoid size explosion)
  - Bridges the gap between theory and implementation
  - Provides convenient technical tool for simulating abstract machines

1. Explicit Substitutions
   - Motivation
   - Two Examples
   - ES - Assesment
2. Normalisation for Calculi with ES
3. Additional Problems in Explicit Substitutions

λx

λx-terms

$$\begin{array}{rcll} M & ::= & x & \text{variable} \\ & | & M\,N & \text{application} \\ & | & \lambda x.M & \text{abstraction} \\ & | & M[x/N] & \text{closure} \end{array}$$

A term without occurrences of closures is a pure term

λx

Rewrite rules

$$\begin{array}{llcl} Beta: & (\lambda x.M)\, N & \rightarrow & M[x/N] \\ \\ App: & (M\, N)[x/P] & \rightarrow & M[x/P]\, N[x/P] \\ Abs: & (\lambda x.M)[y/P] & \rightarrow & \lambda x.M[y/P] \\ Varx: & x[x/P] & \rightarrow & P \\ Vary: & y[x/P] & \rightarrow & y \end{array}$$

- λx-terms are considered modulo α-equivalence
- x is the TRS given by App, Abs, $Varx$, $Vary$

Simulating the λ-calculus in λx

Lambda calculus reduction step

$$(\lambda x.x\,(y\,x))\,N \rightarrow_\beta N\,(y\,N)$$

λx reduction steps

$$\begin{array}{lll}
\underline{(\lambda x.x\,(y\,x))\,N} & \rightarrow_{Beta} & \underline{(x\,(y\,x))[x/N]} \\
 & \rightarrow_{App} & x[x/N]\,\underline{(y\,x)[x/N]} \\
 & \rightarrow_{App} & x[x/N]\,(\underline{y[x/N]}\,x[x/N]) \\
 & \rightarrow_{Vary} & \underline{x[x/N]}\,(y\,x[x/N]) \\
 & \rightarrow_{Varx} & N\,(y\,\underline{x[x/N]}) \\
 & \rightarrow_{Varx} & N\,(y\,N)
\end{array}$$

Properties of λx (1/6) - Basics

Lemma

1. x is SN
2. The x-normal form of a term is a pure term

Lemma

1. $M \to_{\beta} N$ implies $M \twoheadrightarrow_{\lambda x} N$ (simulation)
2. $M \to_{\lambda x} N$ implies $M \twoheadrightarrow_{\beta} N$ (projection)

Simulation mimicks the β step with a *Beta* step followed by reduction to x normal form

Properties of λx (2/6) - PSN

Lemma (Preservation of SN)

If M is β-SN, then M is λx-SN

- Does not hold for some calculi with ES
- Failure of PSN was main thrust in development of ES

Properties of λx (3/6) - Non-orthogonality

Lemma

λx is not orthogonal (λ-calculus is!)

Proof (counterexample)

The (only) critical pair is

$$
\begin{tikzcd}
 & ((\lambda x.M)\, N)[y/P] \arrow[dl, "Beta"'] \arrow[dr, "App"] & \\
M[x/N][y/P] & & (\lambda x.M)[y/P]\, N[y/P]
\end{tikzcd}
$$

Properties of λx (4/6) - Failure of CR

Lemma

λx is not CR

Proof (counterexample)

$$
\begin{tikzcd}
 & ((\lambda x.M)\, N)[y/P] \arrow[dl, "Beta"'] \arrow[dr, "App"] & \\
M[x/N][y/P] & & (\lambda x.M)[y/P]\, N[y/P] \arrow[d, "Abs"'] \\
 & & (\lambda x.M[y/P])\, N[y/P] \arrow[d, "Beta"'] \\
 & & M[y/P][x/N[y/P]]
\end{tikzcd}
$$

Properties of λx (5/6) - Ground CR

Lemma

λx is CR (on ground terms!)

Proof

$$
\begin{tikzcd}
 & & M \arrow[dll, two heads, "\lambda x"'] \arrow[d, "x"', ->|] \arrow[drr, two heads, "\lambda x"] & & \\
P \arrow[d, "x"', ->|] & & \overline{M} \arrow[dll, two heads, "\beta"'] \arrow[drr, two heads, "\beta"] & & Q \arrow[d, "x", ->|] \\
\overline{P} \arrow[drr, two heads, "\beta"'] & & \textcolor{blue}{\text{CR for } \beta} & & \overline{Q} \arrow[dll, two heads, "\beta"] \\
 & & R & &
\end{tikzcd}
$$

- $\rightarrow\!\!|_x$ denotes x-reduction to normal form
- Interpretation technique (Hardin)

Properties of λx (6/6) - Failure of FC

Full composition (FC): $M[x/P] \twoheadrightarrow_{\lambda x} M\{x/P\}$, for all $M \in \mathcal{T}(\lambda x)$

Lemma

λx does not enjoy FC

Proof (counterexample)

The external closure is blocked in $x[x/y][y/z]$

- λx lacks composition of substitutions
- There are calculi of ES with composition that do not satisfy FC

Summary of Properties[1]

Calculus	**CR**	**PSN**	**Sim**	**FC**
$\lambda\upsilon$[1994] λs λt[1996] λx[1994]	No	Yes	Yes	No
$\lambda\sigma$[1991] $\lambda\sigma_{SP}$[1991]	No	No	Yes	Yes
$\lambda\sigma_{\Uparrow}$[1992]λse[1997]	Yes	No	Yes	Yes
$\lambda\zeta$[1996]	Yes	Yes	No	No
λws [1999]	Yes	Yes	Yes	No
$\lambda\mathtt{lxr}$[2005]	?	Yes	Yes	Yes
λes[2007]	Yes	Yes	Yes	Yes

[1] Source: [Kesner2007]

$\lambda\sigma$

- Another calculus with explicit substitutions
- One of the first to appear [ACCL1991]
- Sparked much work in the area
- The first calculus for which PSN was shown to fail (Melliès, 1995)
- We'll provide a brief comparison with λx shortly

$\lambda\sigma$

Terms come in two sorts

$\lambda\sigma$-terms ($\mathcal{T}(\lambda\sigma)$)

$$
\begin{array}{lllllllll}
M & ::= & \mathbf{1} & \text{index} & \quad & s & ::= & id & \text{identity subst} \\
 & | & M\,N & \text{application} & & & | & \uparrow & \text{shift} \\
 & | & \lambda M & \text{abstraction} & & & | & M \cdot s & \text{cons} \\
 & | & M[s] & \text{closure} & & & | & s \circ t & \text{compose}
\end{array}
$$

Indices $1, 2, 3, \ldots$ are represented as $\mathbf{1}$, $\mathbf{1}[\uparrow]$, $\mathbf{1}[\uparrow \circ \uparrow]$, etc.

$\lambda\sigma$

$$
\begin{array}{llcl}
\mathit{Beta}: & (\lambda M)\, N & \rightarrow & M[N \cdot id] \\
\\
\mathit{App}: & (M\, N)[s] & \rightarrow & M[s]\, N[s] \\
\mathit{Abs}: & (\lambda M)[s] & \rightarrow & \lambda M[\mathbf{1} \cdot (s \circ \uparrow)] \\
\mathit{Clos}: & M[s][t] & \rightarrow & M[s \circ t] \\
\mathit{VarCons}: & \mathbf{1}[M \cdot s] & \rightarrow & M \\
\mathit{VarId}: & \mathbf{1}[id] & \rightarrow & \mathbf{1} \\
\\
\mathit{Map}: & (M \cdot s) \circ t & \rightarrow & M[t] \cdot (s \circ t) \\
\mathit{IdL}: & id \circ s & \rightarrow & s \\
\mathit{Ass}: & (s_1 \circ s_2) \circ s_3 & \rightarrow & s_1 \circ (s_2 \circ s_3) \\
\mathit{ShiftCons}: & \uparrow \circ (M \cdot s) & \rightarrow & s \\
\mathit{ShiftId}: & \uparrow \circ\, id & \rightarrow & \uparrow
\end{array}
$$

"Very" non-orthogonal: It has 11 critical pairs!

Simulating β

$$
\begin{array}{lll}
\underline{(\lambda \mathbf{1}\,(\mathbf{2}\,\mathbf{1}))\,N} & \longrightarrow_{Beta} & \underline{(\mathbf{1}\,(\mathbf{2}\,\mathbf{1}))[N\cdot id]} \\
 & \longrightarrow_{App} & \mathbf{1}[N\cdot id]\,\underline{(\mathbf{2}\,\mathbf{1})[N\cdot id]} \\
 & \longrightarrow_{App} & \mathbf{1}[N\cdot id]\,(\mathbf{2}[N\cdot id]\,\mathbf{1}[N\cdot id]) \\
 & = & \mathbf{1}[N\cdot id]\,(\underline{\mathbf{1}[\uparrow][N\cdot id]}\,\mathbf{1}[N\cdot id]) \\
 & \longrightarrow_{Clos} & \mathbf{1}[N\cdot id]\,(\mathbf{1}\underline{[\uparrow\circ(N\cdot id)]}\,\mathbf{1}[N\cdot id]) \\
 & \longrightarrow_{VarId} & \mathbf{1}[N\cdot id]\,(\underline{\mathbf{1}[id]}\,\mathbf{1}[N\cdot id]) \\
 & \longrightarrow_{ShiftId} & \underline{\mathbf{1}[N\cdot id]}\,(\mathbf{1}\,\mathbf{1}[N\cdot id]) \\
 & \longrightarrow_{VarCons} & N\,(\mathbf{1}\,\underline{\mathbf{1}[N\cdot id]}) \\
 & \longrightarrow_{VarCons} & N\,(\mathbf{1}\,N)
\end{array}
$$

Failure of PSN

Prop.

PSN fails for $\lambda\sigma$

Proof

[Melliès1995] exhibits a (typed, hence β-SN!) term $M \in \mathcal{T}(\lambda)$ which admits an infinite $\lambda\sigma$-reduction sequence.

$$
\begin{array}{lll}
\lambda((\lambda(\lambda\mathbf{1})\,((\lambda\mathbf{1})\,\mathbf{1}))\,((\lambda\mathbf{1})\,\mathbf{1})) & \rightarrow_{Beta} & \lambda((\lambda\mathbf{1}[((\lambda\mathbf{1})\,\mathbf{1})\cdot id])\,((\lambda\mathbf{1})\,\mathbf{1})) \\
 & \rightarrow_{Beta} & \lambda\mathbf{1}[((\lambda\mathbf{1})\,\mathbf{1})\cdot id][((\lambda\mathbf{1})\,\mathbf{1})\cdot id] \\
 & = & \lambda\mathbf{1}[s_1][s_1] \\
 & \rightarrow_{Clos} & \lambda\mathbf{1}[s_1 \circ s_1]
\end{array}
$$

He then shows that $s_1 \circ s_1$ can (roughly) bury a copy of itself inside an evergrowing context

λx vs $\lambda\sigma$

λx

★	simple formulation
★	helped devise interesting techiques for proving PSN [Bloo1997]
X	Not first-order TRS
X	Too simple!

$\lambda\sigma$

★	first-order TRS
★	allows composition of substitution
X	complex dynamics
X	PSN fails

"This [Melliès' counterexample] was shocking news at the time, and a gang of ill-advised researchers decided that the $\lambda\sigma$-calculus should be "cut down" to a simpler calculus (like λx) where substitutions are unary, and cannot be composed together. But this was like killing all the beauty and interest of the system.", Anonymous referee report (Dec.2003)

λx vs $\lambda\sigma$

λx

- ★ simple formulation
- ★ helped devise interesting techiques for proving PSN [Bloo1997]
- X Not first-order TRS
- X Too simple!

$\lambda\sigma$

- ★ first-order TRS
- ★ allows composition of substitution
- X complex dynamics
- X PSN fails

"*This [Melliès' counterexample] was shocking news at the time, and a gang of ill-advised researchers decided that the $\lambda\sigma$-calculus should be "cut down" to a simpler calculus (like λx) where substitutions are unary, and cannot be composed together. But this was like killing all the beauty and interest of the system.*", Anonymous referee report (Dec.2003)

1. Explicit Substitutions
   - Motivation
   - Two Examples
   - ES - Assesment
2. Normalisation for Calculi with ES
3. Additional Problems in Explicit Substitutions

ES - Assesment

- Calculi of ES are complex

"*Trying to keep track of ES is like trying to keep track of a box of chicken after letting them loose in the center of Paris*", V.van Oostrom paraphrasing P-A.Melliès

- The principal culprit is that they are non-orthogonal
  - Rewriting theory of non-orthogonal systems is notably underdeveloped in comparison with orthogonal ones
- However, they make an interesting study companion for understanding dynamics of non-orthogonal systems:
  - They are left-linear
  - They are close to λ-calculus (well-known)
  - Many are first-order (well-known)

ES - Assesment

Since Melliès surprising result (failure of PSN for $\lambda\sigma$) three lines of research appeared

1. ES as an implementation technique for Proof Assistants, Functional and Logic Languages, etc.

Relevant topics include:

- abstract machines, weak reduction
- HO unification/matching through FO unification/matching
- optimal reduction, etc.

ES - Assesment

2. Devise calculi with ES that enjoy all the good properties (CR, PSN, Sim, FC)

- This is (was?) the most popular
- A plethora of calculi generated in a frantic race against time
- The first acceptable solution: λws [Guillaume and David, 1999]
- State of the art: [KL2005,Kesner2007]

ES - Assesment

3. Develop a theory of normalization for $\lambda\sigma$ (better still, for arbitrary non-orthogonal systems)

- Developed by P-A. Melliès (as seen from yesterday's talk)
- Can be applied to $\lambda\sigma$ [Melliès2000] (today's talk)
- Can be applied to ES calculi arising from arbitrary (orthogonal, pattern) higher-order rewrite systems [Bonelli2005]

1. Explicit Substitutions

2. Normalisation for Calculi with ES
   - Projecting Standard $\lambda\sigma$ Derivations
   - Finite Normalisation Cones for $\lambda\sigma$

3. Additional Problems in Explicit Substitutions

Projecting Standard $\lambda\sigma$ Derivations

- We prove that $\lambda\sigma$ enjoys finite normalisation cones (FNC) for every closed term
- For that we require an intermediate result: the Std-Projection Proposition
- We first take a look at this proposition and then consider FNC

Projection of $\lambda\sigma$ Derivations

Projection of a derivation d in $\lambda\sigma$ to a derivation $\sigma(d)$ in λ-calculus

$$
\begin{tikzcd}
d : M_1 \arrow[r, "\lambda\sigma"'] \arrow[d, "\sigma"'] & M_2 \arrow[r, "\lambda\sigma"'] \arrow[d, "\sigma"'] & M_3 \arrow[r, "\lambda\sigma"'] \arrow[d, "\sigma"'] & \cdots \arrow[r, "\lambda\sigma"'] & M_n \arrow[d, "\sigma"'] \\
\sigma(d) : \sigma(M_1) \arrow[r, "="', "\beta"'] & \sigma(M_2) \arrow[r, "="', "\beta"'] & \sigma(M_3) \arrow[r, "="', "\beta"'] & \cdots \arrow[r, "="', "\beta"'] & \sigma(M_n)
\end{tikzcd}
$$

Q: If d is standard, is $\sigma(d)$ standard?

Projection of $\lambda\sigma$ Derivations

Q: If d is standard, is $\sigma(d)$ standard?
A: Not necessarily

$$d : ((\lambda(\mathbf{1}\mathbf{1}))\mathbf{1})[\underline{(\lambda\mathbf{1})\mathbf{1}} \cdot id] \quad \to_{Beta} \quad \underline{((\lambda(\mathbf{1}\mathbf{1}))\mathbf{1})}[\mathbf{1}[\mathbf{1} \cdot id] \cdot id]$$
$$\to_{Beta} \quad (\mathbf{1}\mathbf{1})[\mathbf{1} \cdot id][\mathbf{1}[\mathbf{1} \cdot id] \cdot id]$$

$$\sigma(d) : (\lambda(\mathbf{1}\mathbf{1}))\underline{((\lambda\mathbf{1})\mathbf{1})} \quad \to_{\beta} \quad \underline{(\lambda(\mathbf{1}\mathbf{1}))\mathbf{1}}$$
$$\to_{\beta} \quad \mathbf{1}\,\mathbf{1}$$

Disjoint *Beta* redexes become nested β redexes after they are projected

Q: What if d ends in a σ-normal form?
A: Then yes!

Projection of $\lambda\sigma$ Derivations

Q: If d is standard, is $\sigma(d)$ standard?

A: Not necessarily

$$
\begin{array}{rll}
d : ((\lambda(\mathbf{1}\mathbf{1}))\mathbf{1})[\underline{(\lambda\mathbf{1})\mathbf{1}} \cdot id] & \rightarrow_{Beta} & \underline{((\lambda(\mathbf{1}\mathbf{1}))\mathbf{1})}[\mathbf{1}[\mathbf{1} \cdot id] \cdot id] \\
 & \rightarrow_{Beta} & (\mathbf{1}\mathbf{1})[\mathbf{1} \cdot id][\mathbf{1}[\mathbf{1} \cdot id] \cdot id]
\end{array}
$$

$$
\begin{array}{rll}
\sigma(d) : (\lambda(\mathbf{1}\mathbf{1}))\underline{((\lambda\mathbf{1})\mathbf{1})} & \rightarrow_{\beta} & \underline{(\lambda(\mathbf{1}\mathbf{1}))\mathbf{1}} \\
 & \rightarrow_{\beta} & \mathbf{1}\,\mathbf{1}
\end{array}
$$

Disjoint *Beta* redexes become nested β redexes after they are projected

Q: What if d ends in a σ-normal form?

A: Then yes!

Projection of $\lambda\sigma$ Derivations Ending in σ-nf

Q: How does this preclude the previous counterexample?
A: Consequence of

1. d ends in σ-normal form implies all ES are computed
2. reduction inside left argument of $M \cdot s$ cannot create redex above it (hence whatever is done in M can be permuted with what is done outside M)

Eg. the following cannot happen if d is a standard $\lambda\sigma$-derivation and N is a σ-nf

$$d : M[(\underline{(\lambda\mathbf{1})\,\mathbf{1}}) \cdot id] \rightarrow_{Beta} M[\mathbf{1}[\mathbf{1} \cdot id] \cdot id] \twoheadrightarrow N$$

Std-Projection

Prop. (Std-Projection)

1. $d : M \twoheadrightarrow N$ standard in $\lambda\sigma$ with N in σ-normal form

$$\Longrightarrow$$

$\sigma(d) : \sigma(M) \twoheadrightarrow N$ standard in λ-calculus

2. Moreover, each *Beta* redex in d is projected to a unique β redex in $\sigma(d)$

We make use of this result in our proof on FNC for $\lambda\sigma$

1. Explicit Substitutions

2. Normalisation for Calculi with ES
   - Projecting Standard $\lambda\sigma$ Derivations
   - Finite Normalisation Cones for $\lambda\sigma$

3. Additional Problems in Explicit Substitutions

Finite Normalisation Cones

Recall from yesterday

A normalisation cone from M is a set $\{e_i^M : M \twoheadrightarrow P_i\}$ of normalising derivations s.t. for each normalising derivation $f : M \twoheadrightarrow N$, there exists a unique i, $f \equiv e_i$.

A TRS enjoys finite normalisation cones (FNC) when for any M there exists a finite normalisation cone for M.

Finite Normalisation Cones for $\lambda\sigma$

Thm

Every closed $\lambda\sigma$-term enjoys finite normalisation cones

Proof

By contradiction and in three steps.

1. **Step 1**: König's Lemma
2. **Step 2**: Strong σ-normalisation
3. **Step 3**: Std-Projection

Finite Normalisation Cones for $\lambda\sigma$

Proof (cont.)

Step 1 Suppose $\lambda\sigma$ does not enjoy FNC on closed terms (i.e. there is a closed term M_1 with an infinite no. of $\equiv$-distinct normalisation derivations $M_1 \twoheadrightarrow N$).
By König deduce the existence of an infinite $\lambda\sigma$ derivation d_∞ from M_1

$$\begin{tikzcd}
d_\infty : M_1 \arrow[r] \arrow[dr, two heads] & M_2 \arrow[r] \arrow[d, two heads] & M_3 \arrow[r] \arrow[dl, two heads] & \cdots \arrow[dll, two heads] \\
& N & &
\end{tikzcd}$$

where each $d_i : M_1 \twoheadrightarrow M_i \twoheadrightarrow N$ is standard and normalising
Step 2
Note d_∞ has an infinite number of *Beta*-steps since σ is SN

Finite Normalisation Cones for $\lambda\sigma$

Proof (cont.)

Step 3

- Project each $d_i : M_1 \twoheadrightarrow M_i \twoheadrightarrow N$

$$\begin{tikzcd}
\sigma(d_\infty) : \sigma(M_1) \arrow[r, "="', "\beta"'] \arrow[drr, two heads] & \sigma(M_2) \arrow[r, "=", "\beta"'] \arrow[dr, two heads] & \sigma(M_3) \arrow[r, "=", "\beta"'] \arrow[d, two heads] & \cdots \arrow[dl, two heads] \\
& & \sigma(N) &
\end{tikzcd}$$

  where each $\sigma(d_i) : \sigma(M_1) \twoheadrightarrow \sigma(M_i) \twoheadrightarrow \sigma(N) = N$ is standard (by Std-Projection) and normalising
- Since standard derivations are unique in λ, $\forall i, j, \sigma(d_i) \simeq \sigma(d_j)$. Thus $\forall i, j, |\sigma(s_i)| = |\sigma(d_j)|$
- We reach a contradiction with Std-Projection: $\forall i \exists j, |\sigma(d_j)| > |\sigma(d_i)|$

Sample Strategy

Let $M \in \mathcal{T}(\lambda\sigma)$; consider $\diamond$ a fresh constant. Define vertebra(M) as

1. $\text{vertebra}(\mathbf{1}) = \mathbf{1}$
2. $\text{vertebra}(\lambda P) = \lambda\text{vertebra}(P)$
3. $\text{vertebra}(P\,Q) = \text{vertebra}(P)\diamond$
4. $\text{vertebra}(P[s]) = \text{vertebra}(P)[\diamond]$

Define spine(M) as

1. vertebra(M), if M is not of the form $\lambda\ldots\lambda(\mathbf{i}\,M_1\ldots M_n)$
2. $\lambda\ldots\lambda(\mathbf{i}\,\text{vertebra}(M_1)\ldots\text{vertebra}(M_n))$, when $M = \lambda\ldots\lambda(\mathbf{i}\,M_1\ldots M_n)$

Sample Strategy

A spine redex is a $\lambda\sigma$-redex $r : M \rightarrow N$ whose occurrence is not replaced by the constant $\diamond$ in spine(M)

Lemma

Every spine redex is needed

Sample Strategy

Let $s = \mathbf{2} \cdot id$

$$
\begin{tikzcd}
(\lambda(\lambda\mathbf{1})\,\mathbf{1})\,\mathbf{2} \arrow[d, "Beta"'] & & & \\
((\lambda\mathbf{1})\,\mathbf{1})[s] \arrow[r, "App"'] \arrow[d, "Beta"'] & (\lambda\mathbf{1})[s]\mathbf{1}[s] \arrow[rr, "Abs"'] & & (\lambda\mathbf{1}[\mathbf{1} \cdot (\uparrow \circ s)])\,\mathbf{1}[s] \arrow[d, "Beta"'] \\
\mathbf{1}[\mathbf{1} \cdot id][s] \arrow[d, "Comp"'] \arrow[dd, bend right] \arrow[ddr, "VarCons"] & & & \mathbf{1}[\mathbf{1} \cdot (\uparrow \circ s)][\mathbf{1}[s] \cdot id] \arrow[dl, "VarCons"'] \arrow[dd, "Comp"'] \\
\mathbf{1}[(\mathbf{1} \cdot id) \circ s] \arrow[d, "Map"'] & & \mathbf{1}[\mathbf{1}[s] \cdot id] \arrow[dl, "VarCons"'] & \\
\mathbf{1}[\mathbf{1}[s] \cdot (id \circ s)] \arrow[r, "VarCons"'] & \mathbf{1}[s] \arrow[r, "VarCons"'] & \mathbf{2} & \cdots
\end{tikzcd}
$$

Credits

- FNC for $\lambda\sigma$ was proved in [Melliès1996,2000]
- Except for the proof of Std-Projection, the theory relies on diagrammatic techniques
- This yields a general theory applicable to many classes of rewrite systems

1. Explicit Substitutions
2. Normalisation for Calculi with ES
3. Additional Problems in Explicit Substitutions

Assorted Problems in Explicit Substitutions

- ES Without Critical Pairs: develop a theory of treks [Melliès2002] for ES
- Axiomatic Rewriting: Extend work of Mellies to ES
  - Using the "old" axiomatics based on nesting [Melliès1996]
  - Using the newer diagrammatic formulation [Melliès2005]
- Infinitary Rewriting: Infinitary lambda calculus with ES
  - If rewrite step=finite computation, then it makes sense to replace substitution on infinite terms with ES
- Matching modulo superdevelopments: Lambda calculus with ES
  - Higher-order matching where restriction is put on reduction rather than the terms
  - Notion of residual must be revisited

Thank you for inviting me!