A Confluent Pattern Calculus with Hedge Variables

Sandra Alves[1] Besik Dundua[1,3] Mário Florido[1]
Temur Kutsia[2]

DCC-FC & LIACC, University of Porto, Portugal

RISC, Johannes Kepler University, Linz, Austria

VIAM, Ivane Javakhishvili Tbilisi State University, Georgia

Outline

Pattern Calculus

- Pattern calculi extend the lambda calculus with patterns.
- λ abstracts not only variables but also terms.
- Pattern calculi integrate pattern matching capabilities into the λ-calculus.
- Pattern calculi are expressive, but in general the confluence property is lost.
- To recover confluence, some restrictions on patterns and their applications are imposed.

Pattern Calculus

- Lambda Calculus with Patterns was introduced by van Oostrom in 1990.
- Since then various formalisms that address integration of pattern matching capabilities with the lambda calculus have been investigated.
- In 2007, Cirstea and Faure proposed a generic confluence proof for the dynamic pattern calculus.
- The calculus is parametrized by a function that defines the unitary matching algorithm. There are some conditions the function should satisfy, in order the guarantee the confluence.
- We extended the dynamic pattern calculus with hedge variables and studied conditions that should be satisfied by the function that defines the finitary matching.

Terms

- $M, N ::= x \mid\ f \mid (M\,N) \mid (M\,X) \mid \lambda_{\mathcal{V}} M.N \mid M + N$

where

- x is a term variable
- X is a hedge variable
- f is a constant.
- $(M\,N)$ is an application of a term to a term
- $(M\,X)$ is an application of a term to a hedge variable

Terms

Defined by the grammar:

$$M, N ::= x \mid \; f \mid (M\, N) \mid (M\, X) \mid \lambda_{\mathcal{V}} M.N \mid M + N$$

- $\lambda_{\mathcal{V}} M.N$ is an abstraction where the term M is called a pattern.
- $\mathcal{V}$ is a subset of the set of free variables of M, representing the set of variables bound by the abstraction.

Terms

Defined by the grammar:

$$M, N ::= x \mid \; f \mid (M\,N) \mid (M\,X) \mid \lambda_{\mathcal{V}} M.N \mid M + N$$

- $\lambda_{\mathcal{V}} M.N$ is an abstraction where the term M is called a pattern.
- $\mathcal{V}$ is a subset of the set of free variables of M, representing the set of variables bound by the abstraction.
- For example, a term $\lambda_{\{x,X\}} fxYX.gXyY$ has bound variables x, X and free variables y, Y.

Terms

Defined by the grammar:

$$M, N ::= x \mid \; f \mid (M\,N) \mid (M\,X) \mid \lambda_{\mathcal{V}} M.N \mid M + N$$

- $\lambda_{\mathcal{V}} M.N$ is an abstraction where the term M is called a pattern.
- $\mathcal{V}$ is a subset of the set of free variables of M, representing the set of variables bound by the abstraction.
- For example, a term $\lambda_{\{x,X\}} fxYX.gXyY$ has bound variables x, X and free variables y, Y.
- $+$ is a associative, commutative, and idempotent. Moreover, application distributes over $+$ both from the left and from the right. We write ACID for this property.

ACID Normal Form

We work with terms in the ACID normal form with respect to $+$ and application.

Example

- A term not in the ACID normal form

$$\lambda_{\{x,X\}}(fx+gx)X.\, fx(gx+gX+gx)$$

- A term in the ACID normal form

$$\lambda_{\{x,X\}}(fxX+gxX).\, (fx(gx)+fx(gX))$$

Hedges and Substitution

- Hedges are finite (possible empty) sequences of terms and hedge variables.
- Notation: h for hedges. ϵ for the empty hedge.
- For readability, we put hedges in angle brackets if they have more than one element, e.g., $\langle M, X, N \rangle$.

Hedges and Substitution

- Hedges are finite (possible empty) sequences of terms and hedge variables.
- Notation: h for hedges. ϵ for the empty hedge.
- For readability, we put hedges in angle brackets if they have more than one element, e.g., $\langle M, X, N \rangle$.
- A substitution is a mapping from term variables to terms, and from hedge variables to hedges, such that all but finitely many term and hedge variables are mapped to themselves.

Hedges and Substitution

- Hedges are finite (possible empty) sequences of terms and hedge variables.
- Notation: h for hedges. ϵ for the empty hedge.
- For readability, we put hedges in angle brackets if they have more than one element, e.g., $\langle M, X, N \rangle$.
- A substitution is a mapping from term variables to terms, and from hedge variables to hedges, such that all but finitely many term and hedge variables are mapped to themselves.
- Notation: σ and ϑ for substitutions.

Hedges and Substitution

- Hedges are finite (possible empty) sequences of terms and hedge variables.
- Notation: h for hedges. ϵ for the empty hedge.
- For readability, we put hedges in angle brackets if they have more than one element, e.g., $\langle M, X, N \rangle$.
- A substitution is a mapping from term variables to terms, and from hedge variables to hedges, such that all but finitely many term and hedge variables are mapped to themselves.
- Notation: σ and ϑ for substitutions.
- The composition is defined in the standard way.

Substitution Application

Term: $$M = \lambda_{\{x,Y\}}\, \overbrace{f X x Y}^{P} \,.\, \overbrace{y(gX)xZ}^{N}$$

Substitution: $$\sigma = \{x \mapsto gx, y \mapsto \lambda_x x.fxa, Z \mapsto \epsilon, \\ X \mapsto \langle \lambda_x x.x, \lambda_x x.(x + fx)\rangle\}$$

$$M\sigma = \lambda_{\{x',Y\}}\, \overbrace{f(\lambda_x x.x)(\lambda_x x.(x + fx))x' Y}^{P\sigma} \,. \\ \overbrace{(\lambda_x x.fxa)(g(\lambda_x x.x)(\lambda_x x.(x + fx)))x'}^{N\sigma}$$

Matching Equation and Solution

Equation: $fXxY \ll^{?} fabcde$

Solutions:

$$\sigma_1 = \{X \to \epsilon, x \to a, Y \to \langle b, c, d, e\rangle\}$$
$$\sigma_2 = \{X \to a, x \to b, Y \to \langle c, d, e\rangle\}$$
$$\sigma_3 = \{X \to \langle a, b\rangle, x \to c, Y \to \langle d, e\rangle\}$$
$$\sigma_4 = \{X \to \langle a, b, c\rangle, x \to d, Y \to e\}$$
$$\sigma_5 = \{X \to \langle a, b, c, d\rangle, x \to e, Y \to \epsilon\}$$

Operational Semantics

Sol: A function which takes a pattern matching equation and returns a set of solutions.

Operational Semantics

Sol: A function which takes a pattern matching equation and returns a set of solutions.

$$\begin{array}{ll} \beta_{\mathsf{p}}: & (\lambda_{\mathcal{V}} M.N)\, Q \to N\sigma_1 + \cdots + N\sigma_n, \\ & \text{where } N\sigma_1, \ldots, N\sigma_n,\ n \geqslant 1,\ \text{are terms} \\ & Sol(M \ll_{\mathcal{V}} Q) = \{\sigma_1, \ldots, \sigma_n\},\ n \geqslant 1, \\ & M \text{ and } Q \text{ are not of the form } W_1 + W_2. \end{array}$$

Operational Semantics

Sol: A function which takes a pattern matching equation and returns a set of solutions.

$$
\begin{array}{lll}
\beta_{\mathsf{p}}: & & (\lambda_{\mathcal{V}} M.N)\, Q \to N\sigma_1 + \cdots + N\sigma_n, \\
& & \text{where } N\sigma_1, \ldots, N\sigma_n,\ n \geqslant 1, \text{ are terms} \\
& & Sol(M \ll_{\mathcal{V}} Q) = \{\sigma_1, \ldots, \sigma_n\},\ n \geqslant 1, \\
& & M \text{ and } Q \text{ are not of the form } W_1 + W_2. \\
\mathsf{D}_{\mathsf{l}}: & & \lambda_{\mathcal{V}} M_1 + M_2.N \to \lambda_{\mathcal{V}} M_1.N + \lambda_{\mathcal{V}} M_2.N.
\end{array}
$$

Operational Semantics

Sol: A function which takes a pattern matching equation and returns a set of solutions.

$$\begin{array}{ll}
\beta_{\mathsf{p}}: & (\lambda_{\mathcal{V}} M.N)\, Q \to N\sigma_1 + \cdots + N\sigma_n, \\
& \text{where } N\sigma_1, \ldots, N\sigma_n,\ n \geqslant 1, \text{ are terms} \\
& Sol(M \ll_{\mathcal{V}} Q) = \{\sigma_1, \ldots, \sigma_n\},\ n \geqslant 1, \\
& M \text{ and } Q \text{ are not of the form } W_1 + W_2. \\
& \\
\mathsf{D}_{\mathsf{l}}: & \lambda_{\mathcal{V}} M_1 + M_2.N \to \lambda_{\mathcal{V}} M_1.N + \lambda_{\mathcal{V}} M_2.N. \\
& \\
\mathsf{D}_{\mathsf{r}}: & \lambda_{\mathcal{V}} M.N_1 + N_2 \to \lambda_{\mathcal{V}} M.N_1 + \lambda_{\mathcal{V}} M.N_2.
\end{array}$$

Operational Semantics

Sol: A function which takes a pattern matching equation and returns a set of solutions.

$$
\begin{array}{ll}
\beta_{\mathsf{p}}: & (\lambda_{\mathcal{V}} M.N)\, Q \to N\sigma_1 + \cdots + N\sigma_n, \\
& \text{where } N\sigma_1, \ldots, N\sigma_n,\ n \geqslant 1, \text{ are terms} \\
& \mathit{Sol}(M \ll_{\mathcal{V}} Q) = \{\sigma_1, \ldots, \sigma_n\},\ n \geqslant 1, \\
& M \text{ and } Q \text{ are not of the form } W_1 + W_2.
\end{array}
$$

$$
\mathsf{D_l}: \quad \lambda_{\mathcal{V}} M_1 + M_2.N \to \lambda_{\mathcal{V}} M_1.N + \lambda_{\mathcal{V}} M_2.N.
$$

$$
\mathsf{D_r}: \quad \lambda_{\mathcal{V}} M.N_1 + N_2 \to \lambda_{\mathcal{V}} M.N_1 + \lambda_{\mathcal{V}} M.N_2.
$$

Pattern reduction $\to_P$ is a compatible closure of the union of relations $\beta_{\mathsf{p}}, \mathsf{D_l}$ and $\mathsf{D_r}$.

Example: No Confluence

$$
\begin{tikzcd}[column sep=small]
 & & (\lambda_Z fZ.(\lambda_{\{X,Y\}} fXY.fX) fZ) fab \arrow[dl] \arrow[dr] & & \\
 & (\lambda_Z fZ.(f + fZ)) fab \arrow[dl] & & (\lambda_{\{X,Y\}} fXY.fX) fab \arrow[dr] & \\
 f + fab & & & & f + fa + fab
\end{tikzcd}
$$

The example shows that we do not have confluence in general!

How to Obtain Confluence

Goal: Impose restrictions on *Sol* to guarantee confluence.

Sufficient Conditions for Confluence

Condition 1: Preservation of Free Variables

$$\sigma \in \mathit{Sol}(P \ll_{\mathcal{V}} M) \qquad \text{implies} \qquad \begin{cases} \mathit{Dom}(\sigma) = \mathcal{V} \\ \mathtt{fv}(\mathit{Ran}(\sigma)) \subseteq \mathtt{fv}(M) \end{cases}$$

Example

- Assume that the term $(\lambda_{\mathcal{V}} P.N)\, M$ reduces to the term $N\sigma_1 + \cdots + N\sigma_n$ with $\mathit{Sol}(P \ll_{\mathcal{V}} M) = \{\sigma_1, \ldots, \sigma_n\}$.
- Then the inclusion $\mathtt{fv}(N\sigma_i) \subseteq (\lambda_{\mathcal{V}} P.N)\, M$ should hold for any σ_i, $0 < i \leqslant n$.

Sufficient Conditions for Confluence

Condition 2: Stability by Substitution

$$Sol(P \ll_{\mathcal{V}} M) = \overline{\sigma} \text{ implies } \begin{cases} \forall\theta \text{ s.t. } Var(\vartheta) \cap \mathcal{V} = \varnothing \\ Sol(P\theta \ll_{\mathcal{V}} M\theta) = \overline{\sigma}\theta \end{cases}$$

where $\overline{\sigma} = \{\sigma_1, \ldots, \sigma_n\}$ and $\overline{\sigma}\theta = \{(\sigma_1\theta)|_{\mathcal{V}}, \ldots, (\sigma_n\theta)|_{\mathcal{V}}\}, n \geqslant 1$

Example

Violation of the Stability by Substitution Leads to Non-Confluence:

$$
\begin{tikzcd}
& & (\lambda_Z fZ.(\lambda_{\{X,Y\}} fXY.fX) fZ) fab \arrow[dl] \arrow[dr] & & \\
& (\lambda_Z fZ.(f + fZ)) fab \arrow[dl] & & (\lambda_{\{X,Y\}} fXY.fX) fab \arrow[dr] & \\
f + fab & & & & f + fa + fab
\end{tikzcd}
$$

Sufficient Conditions for Confluence

Condition 3: Stability by Reduction

$$\begin{cases} Sol(P \ll_{\mathcal{V}} M) = \overline{\sigma} \\ P \Rightarrow_P P', \\ M \Rightarrow_P M', \end{cases} \quad \text{implies} \quad \begin{cases} Sol(P' \ll_{\mathcal{V}} M') = \overline{\theta} \\ \forall_{1 \leqslant i \leqslant n} \exists_{1 \leqslant j \leqslant m} \text{ s.t. } \sigma_i \Rightarrow_P \theta_j \\ \forall_{1 \leqslant j \leqslant m} \exists_{1 \leqslant i \leqslant n} \text{ s.t } \sigma_i \Rightarrow_P \theta_j. \end{cases}$$

where $\overline{\sigma} = \{\sigma_1, \dots, \sigma_n\}, n \geqslant 1$ and $\overline{\theta} = \{\theta_1, \dots, \theta_m\}, m \geqslant 1$.

$\Rightarrow_P$ is the parallel reduction (details on the next slide).

Parallel Reduction

s stands for a hedge variable or a term.

$$\frac{}{s \Rightarrow_P s} \qquad \frac{s_1 \Rightarrow_P s_1' \quad \ldots \quad s_n \Rightarrow_P s_n'}{\langle s_1, \ldots, s_n \rangle \Rightarrow_P \langle s_1', \ldots, s_n' \rangle} \qquad \frac{M \Rightarrow_P M' \quad s \Rightarrow_P s'}{M\, s \Rightarrow_P M'\, s'}$$

$$\frac{M \Rightarrow_P M' \quad N \Rightarrow_P N'}{\lambda_{\mathcal{V}} M.N \Rightarrow_P \lambda_{\mathcal{V}} M'.N'} \qquad \frac{M_1 \Rightarrow_P M_1' \quad M_2 \Rightarrow_P M_2' \quad N \Rightarrow_P N'}{\lambda_{\mathcal{V}}(M_1 + M_2).N \Rightarrow_P \lambda_{\mathcal{V}} M_1'.N' + \lambda_{\mathcal{V}} M_2'.N'}$$

$$\frac{M \Rightarrow_P M' \quad N \Rightarrow_P N'}{M + N \Rightarrow_P M' + N'} \qquad \frac{M \Rightarrow_P M' \quad N_1 \Rightarrow_P N_1' \quad N_2 \Rightarrow_P N_2'}{\lambda_{\mathcal{V}} M.(N_1 + N_2) \Rightarrow_P \lambda_{\mathcal{V}} M'.N_1' + \lambda_{\mathcal{V}} M'.N_2'}$$

$$\frac{M \Rightarrow_P M' \quad N \Rightarrow_P N' \quad Q \Rightarrow_P Q'}{(\lambda_{\mathcal{V}} M.N)Q \Rightarrow_P N'\sigma_1 + \cdots + N'\sigma_n} \text{ where } Sol(M' \ll_{\mathcal{V}} Q') = \{\sigma_1, \ldots, \sigma_n\}$$

Definition of parallel reduction is extended to substitutions having the same domain by setting $\theta \Rightarrow_P \theta'$ if for all $v \in Dom(\theta) = Dom(\theta')$, we have $v\theta \Rightarrow_P v\theta'$.

Example

Violation of the Stability by Reduction Leads to Non-Confluence.

$$
\begin{tikzcd}[column sep=small]
& (\lambda_{\{x,Y\}} \overbrace{xY}^{P} .x) \overbrace{((\lambda_z z.z)a)}^{M} \arrow[dl, "\{x \to (\lambda_z z.z),\, Y \to a\}"'] \arrow[dr] & & \\
(\lambda_z z.z) & & (\lambda_{\{x,Y\}} \overbrace{xY}^{P} .x)a \arrow[dr, "\{x \to a,\, Y \to \epsilon\}"] & \\
& & & a
\end{tikzcd}
$$

Confluence

Theorem

The pattern calculus with hedge variables where Sol satisfies preservation of free variables, stability by substitution and stability by reduction properties is confluent.

Matching with Hedge Variables

We can define $Sol(P \ll_{\mathcal{V}} M)$ as a partial function with the following conditions:

- If P contains a λ-abstraction or a $+$, or if $P \mp x$ and M contains a λ-abstraction or a $+$ or a hedge variables, or if $\mathtt{fv}(P) \mp \mathcal{V}$, then undifiend.
- Otherwise, $Sol(P \ll_{\mathcal{V}} M)$ normalizes the matching problem $P \lll^{?} M$ with respect to following rules and collects substitutions σ from the success states.

$$M \lll^{?} M \rightsquigarrow_{\varepsilon} \varnothing.$$

$$P_1\, P_2 \lll^{?} M_1\, M_2 \rightsquigarrow_{\varepsilon} \{P_1 \lll^{?} M_1, P_2 \lll^{?} M_2\}$$

$$x \lll^{?} M \rightsquigarrow_{\{x \mapsto M\}} \varnothing.$$

$$P\, X \lll^{?} M s_1 \cdots s_n\, s'_1 \cdots s'_m \rightsquigarrow_{\{X \mapsto \langle s'_1, \ldots, s'_m \rangle\}} \{P \lll^{?} M s_1 \cdots s_n\}.$$

Concluding Remarks

- We integrated hedge variables in the pattern calculus.

Concluding Remarks

- We integrated hedge variables in the pattern calculus.
- Studied operational semantics of the derived calculus, parametrized by the function *Sol* for finitary matching.

Concluding Remarks

- We integrated hedge variables in the pattern calculus.
- Studied operational semantics of the derived calculus, parametrized by the function *Sol* for finitary matching.
- Imposed conditions on the *Sol* function under which the calculus is confluent.

Concluding Remarks

- We integrated hedge variables in the pattern calculus.
- Studied operational semantics of the derived calculus, parametrized by the function *Sol* for finitary matching.
- Imposed conditions on the *Sol* function under which the calculus is confluent.
- A concrete example of *Sol* which satisfies those conditions is hedge matching.

Work in Progress

- Relaxing conditions for the *Sol* function under which confluence is guaranteed.
- Introduction of types and studying properties such as subject reduction, strong normalization, etc.