Principal Typings for Explicit Substitution*

Daniel Lima Ventura & Mauricio Ayala-Rincón
Departamento de Matemática - UnB

Fairouz Kamareddine
ULTRA Group - Heriot-Watt University

4^{th} International Workshop on Higher-Order Rewriting

June 25, 2007, Paris

* Research supported by the Brazilian Research Council - CNPq

Universidade de Brasília

Outline

Principal Typing

Let $\Gamma \vdash b : A$ be a type judgement in some type system S

- $\tau = (\Gamma, A)$ is a typing of b in S ($S \blacktriangleright b : \tau$).

Principal Typing

Let $\Gamma \vdash b : A$ be a type judgement in some type system S

- $\tau = (\Gamma, A)$ is a typing of b in S ($S \blacktriangleright b : \tau$).
- τ is a **principal typing** (PT) of b if $S \blacktriangleright b : \tau$ and τ represents any other possible typing of b.

Principal Typing

Let $\Gamma \vdash b : A$ be a type judgement in some type system S

- $\tau = (\Gamma, A)$ is a typing of b in S ($S \blacktriangleright b : \tau$).
- τ is a **principal typing** (PT) of b if $S \blacktriangleright b : \tau$ and τ represents any other possible typing of b.
- PT property allows *compositional* type inference

Principal Typing vs. Principal Type [Jim96]

Given term b and context Γ, A is a **principal type** of b if it represents any other possible type of b in Γ.

Principal Typing vs. Principal Type [Jim96]

Given term b and context Γ, A is a **principal type** of b if it represents any other possible type of b in Γ.

Principal Type: $\Gamma \vdash b :?$

Principal Typing vs. Principal Type [Jim96]

Given term b and context Γ, A is a **principal type** of b if it represents any other possible type of b in Γ.

Principal Type: $\Gamma \vdash b :?$

Example

Question: $y{:}A{\to}A \vdash \lambda_x.(y\ x) :?$

Principal Typing vs. Principal Type [Jim96]

Given term b and context Γ, A is a **principal type** of b if it represents any other possible type of b in Γ.

Principal Type: $\Gamma \vdash b :?$

Example

Question: $y{:}A{\to}A \vdash \lambda_x.(y\ x) :?$
Answer: $A{\to}A$

Principal Typing vs. Principal Type [Jim96]

Given term b and context Γ, A is a **principal type** of b if it represents any other possible type of b in Γ.

Principal Type: $\Gamma \vdash b :?$

Example

Question: $y{:}A{\rightarrow}A \vdash \lambda_x.(y\ x) :?$
Answer: $A{\rightarrow}A$

Question: $y{:}A{\rightarrow}B \vdash \lambda_x.(y\ x) :?$

Principal Typing vs. Principal Type [Jim96]

Given term b and context Γ, A is a **principal type** of b if it represents any other possible type of b in Γ.

Principal Type: $\Gamma \vdash b :?$

Example

Question: $y{:}A{\to}A \vdash \lambda_x.(y\ x) :?$
Answer: $A{\to}A$

Question: $y{:}A{\to}B \vdash \lambda_x.(y\ x) :?$
Answer: $A{\to}B$

Principal Typing vs. Principal Type

Principal Typing: $? \vdash b :?$

Principal Typing vs. Principal Type

Principal Typing: $? \vdash b :?$

Example

Question: $? \vdash \lambda_x.(y\ x) :?$

Principal Typing vs. Principal Type

Principal Typing: $? \vdash b :?$

Example

Question: $? \vdash \lambda_x.(y\ x) :?$
Answer: $(y{:}A{\rightarrow}B, A{\rightarrow}B)$

Principal Typing vs. Principal Type

Principal Typing: $? \vdash b :?$

Example

Question: $? \vdash \lambda_x.(y\ x) :?$
Answer: $(y{:}A{\rightarrow}B, A{\rightarrow}B)$

Question: $? \vdash \lambda_x.(y\ (y\ x)) :?$

Principal Typing vs. Principal Type

Principal Typing: $? \vdash b :?$

Example

Question: $? \vdash \lambda_x.(y\ x) :?$
Answer: $(y{:}A{\rightarrow}B, A{\rightarrow}B)$

Question: $? \vdash \lambda_x.(y\ (y\ x)) :?$
Answer: $(y{:}A{\rightarrow}A, A{\rightarrow}A)$

Principal Typing vs. Principal Type

	Principal Type	Principal Typing
STLC	✓ [Hi97]	✓ [Wells02]
Hindley/Milner	✓ [DM82]	X [Wells02]
System F	?	X [Wells02]
System $\mathbb{I}$	✓ [KW04]	✓ [KW04]

Principal Typing

Definition

For some typing τ in S let $Terms_S(\tau) = \{a | S \blacktriangleright a{:}\tau\}$.

Principal Typing

Definition

For some typing τ in S let $Terms_S(\tau)=\{a|S \blacktriangleright a{:}\tau\}$.

Definition (Typing's Partial Order)

Let $\tau \leq_S \tau'$ iff $Terms_S(\tau) \subseteq Terms_S(\tau')$

Principal Typing

Definition (Wells' PT[Wells02])

A typing τ in system S is principal for some term a iff $S \blacktriangleright a{:}\tau$ and $S \blacktriangleright a{:}\tau'$ implies $\tau \leq_S \tau'$.

Principal Typing

Definition (Wells' PT[Wells02])

A typing τ in system S is principal for some term a iff $S \blacktriangleright a{:}\tau$ and $S \blacktriangleright a{:}\tau'$ implies $\tau \leq_S \tau'$.

Example

For $\tau_1 = (y{:}A{\rightarrow}B, A{\rightarrow}B)$ and $\tau_2 = (y{:}A{\rightarrow}A, A{\rightarrow}A)$ we have $\mathit{Terms}(\tau_1) \subset \mathit{Terms}(\tau_2)$

Principal Typing for STLC

Definition (Hindley's PT [Wells02])

Let $\tau = (\Gamma, B)$, where $\mathrm{TA}_\lambda \blacktriangleright a : \tau$. τ is principal typing of a iff for any typing $\tau' = (\Gamma', B')$ where $\mathrm{TA}_\lambda \blacktriangleright a : \tau'$, then exists some type substitution s such that $s(\Gamma) \subseteq \Gamma'$ and $s(B) = B'$.

Principal Typing for STLC

Definition (Hindley's PT [Wells02])

Let $\tau = (\Gamma, B)$, where $\mathrm{TA}_\lambda \blacktriangleright a : \tau$. τ is principal typing of a iff for any typing $\tau' = (\Gamma', B')$ where $\mathrm{TA}_\lambda \blacktriangleright a : \tau'$, then exists some type substitution s such that $s(\Gamma) \subseteq \Gamma'$ and $s(B) = B'$.

Theorem ([Wells02])

A typing τ is principal according to Wells' PT iff τ is principal according to Hindley's PT.

Simple Type System

Definition (Simple Types and Contexts)

Types $A ::= K \mid A \rightarrow A$ **Contexts** $\Gamma ::= nil \mid A.\Gamma$

Simple Type System

Definition (Simple Types and Contexts)

Types $A ::= K \mid A \to A$ **Contexts** $\Gamma ::= nil \mid A.\Gamma$

- $|\Gamma|$: Γ length.

Simple Type System

Definition (Simple Types and Contexts)

Types $A ::= K \mid A \to A$ **Contexts** $\Gamma ::= nil \mid A.\Gamma$

- $|\Gamma|$: Γ length.
- For $\Gamma = A_1.A_2.\cdots.A_m$:
  $\Gamma_{\leq n} = A_1.\cdots.A_n$
  $\Gamma_{\geq n} = A_n.\cdots.A_m$

Simple Type System

Definition (Simple Types and Contexts)

Types $A ::= K \mid A \to A$ **Contexts** $\Gamma ::= nil \mid A.\Gamma$

- $|\Gamma|$: Γ length.
- For $\Gamma = A_1.A_2.\cdots.A_m$:
  $\Gamma_{\leq n} = A_1.\cdots.A_n$
  $\Gamma_{\geq n} = A_n.\cdots.A_m$
- $\Gamma_{<n}$ and $\Gamma_{>n}$ defined similar

The System $TA_{\lambda dB}$

Syntax

Terms $\quad a ::= \underline{n} \mid (a\ a) \mid \lambda.a$

Typing Rules

$$A.\Gamma \vdash \underline{1}\ : A \ (Var) \qquad \frac{\Gamma \vdash \underline{n}\ : B}{A.\Gamma \vdash \underline{n+1}\ : B}\ (Varn)$$

$$\frac{A.\Gamma \vdash b : B}{\Gamma \vdash \lambda.b : A \to B}\ (Lambda) \qquad \frac{\Gamma \vdash a : A \to B \quad \Gamma \vdash b : A}{\Gamma \vdash (a\ b) : B}\ (App)$$

PT in $TA_{\lambda dB}$

Lemma (Weakening)

If $\Gamma \vdash a : B$, then $\Gamma.A \vdash a : B$ for any type A.

Definition (PT in $TA_{\lambda dB}$)

Let $\tau = (\Gamma, B)$, where $\mathrm{TA}_{\lambda dB} \blacktriangleright a : \tau$. τ is principal typing of a iff for any typing $\tau' = (\Gamma', B')$ where $\mathrm{TA}_{\lambda dB} \blacktriangleright a : \tau'$, then exists some type substitution s such that $s(\Gamma) = \Gamma'_{\leq|\Gamma|}$ and $s(B) = B'$.

PT for $TA_{\lambda dB}$

Theorem (Correspondence for $TA_{\lambda dB}$)

A typing τ is PT of b in $TA_{\lambda dB}$ iff τ is principal of b according to Wells' PT

PT for $TA_{\lambda dB}$

Theorem (Correspondence for $TA_{\lambda dB}$)

A typing τ is PT of b in $TA_{\lambda dB}$ iff τ is principal of b according to Wells' PT

Theorem (PT for $TA_{\lambda dB}$)

$TA_{\lambda dB}$ satisfies PT property

Type Inference for $TA_{\lambda dB}$ [AyMu2000]

1st Given term a, let a' be its annoted version.
Ex: for $a = \lambda.(\underline{2}\ \ \underline{1})$, $a' = (\lambda.(\underline{2}\,{}^{\Gamma_1}_{A_1}\ \ \underline{1}\,{}^{\Gamma_2}_{A_2})^{\Gamma_3}_{A_3})^{\Gamma_4}_{A_4}$

Type Inference for $TA_{\lambda dB}$ [AyMu2000]

1st Given term a, let a' be its annoted version.
Ex: for $a = \lambda.(\underline{2}\ \ \underline{1})$, $a' = (\lambda.(\underline{2}\,{}^{\Gamma_1}_{A_1}\ \ \underline{1}\,{}^{\Gamma_2}_{A_2})^{\Gamma_3}_{A_3})^{\Gamma_4}_{A_4}$

2nd Let R_0 be the set of subterms of a'. Start using the type inference algorithm on $\langle R_0, \varnothing \rangle$

Type Inference for $TA_{\lambda dB}$ [AyMu2000]

1st Given term a, let a' be its annoted version.
Ex: for $a = \lambda.(\underline{2}\ \ \underline{1})$, $a' = (\lambda.(\underline{2}\,{}^{\Gamma_1}_{A_1}\ \ \underline{1}\,{}^{\Gamma_2}_{A_2})^{\Gamma_3}_{A_3})^{\Gamma_4}_{A_4}$

2nd Let R_0 be the set of subterms of a'. Start using the type inference algorithm on $\langle R_0, \varnothing \rangle$

3rd When $\langle \varnothing, E \rangle$ is reached, E is the set of equations on type and context variables

Type Inference for $TA_{\lambda dB}$ [AyMu2000]

1st Given term a, let a' be its annoted version.
Ex: for $a = \lambda.(\underline{2}\ \ \underline{1})$, $a' = (\lambda.(\underline{2}\,{}^{\Gamma_1}_{A_1}\ \ \underline{1}\,{}^{\Gamma_2}_{A_2})^{\Gamma_3}_{A_3})^{\Gamma_4}_{A_4}$

2nd Let R_0 be the set of subterms of a'. Start using the type inference algorithm on $\langle R_0, \varnothing \rangle$

3rd When $\langle \varnothing, E \rangle$ is reached, E is the set of equations on type and context variables

4th Use a first order unification algorithm to give you the m.g.u.

Type Inference for $TA_{\lambda dB}$

(Var)	$\langle R \cup \{\underline{1}^{\Gamma}_{A}\}, E\rangle \rightarrow$ $\langle R, E \cup \{\Gamma = A.\Gamma'\}\rangle$
(Varn)	$\langle R \cup \{\underline{n}^{\Gamma}_{A}\}, E\rangle \rightarrow$ $\langle R, E \cup \{\Gamma = A'_1. \cdots .A'_{n-1}.A.\Gamma'\}\rangle$
(Lambda)	$\langle R \cup \{(\lambda.a^{\Gamma_1}_{A_1})^{\Gamma_2}_{A_2}\}, E\rangle \rightarrow$ $\langle R, E \cup \{A_2 = A^* \rightarrow A_1, \Gamma_1 = A^*.\Gamma_2\}\rangle$
(App)	$\langle R \cup \{(a^{\Gamma_1}_{A_1}\ b^{\Gamma_2}_{A_2})^{\Gamma_3}_{A_3}\}, E\rangle \rightarrow$ $\langle R, E \cup \{\Gamma_1 = \Gamma_2, \Gamma_2 = \Gamma_3, A_1 = A_2 \rightarrow A_3\}\rangle$

Obs: A', A^* and Γ' are fresh variables

The System $TA_{\lambda s_e}$[KR97]

Syntax

Terms $\quad a ::= \underline{n} \,|\, (a\ a) \,|\, \lambda.a \,|\, a\,\sigma^i a \,|\, \varphi^j_k\, a$

Typing Rules

$$\frac{\Gamma_{\leq k}.\Gamma_{\geq k+i} \vdash a : A}{\Gamma \vdash \varphi^i_k\, a : A}\ (Phi) \qquad \frac{\Gamma_{\geq i} \vdash b : B \quad \Gamma_{<i}.B.\Gamma_{\geq i} \vdash a : A}{\Gamma \vdash a\,\sigma^i b : A}\ (Sigma)$$

PT in $TA_{\lambda s_e}$

Definition (PT in $TA_{\lambda s_e}$)

Let $\tau = (\Gamma, B)$, where $\mathrm{TA}_{\lambda s_e} \blacktriangleright a : \tau$. τ is principal typing of a iff for any typing $\tau' = (\Gamma', B')$ where $\mathrm{TA}_{\lambda s_e} \blacktriangleright a : \tau'$, then exists some type substitution s such that $s(\Gamma) = \Gamma'_{\leq|\Gamma|}$ and $s(B) = B'$.

Theorem (Correspondence for $TA_{\lambda s_e}$)

A typing τ is PT of b in $TA_{\lambda s_e}$ iff τ is principal of b according to Wells' PT definition

PT for $TA_{\lambda s_e}$

Theorem (PT for $TA_{\lambda s_e}$)

$TA_{\lambda s_e}$ satisfies PT property

PT for $TA_{\lambda s_e}$

Theorem (PT for $TA_{\lambda s_e}$)

$TA_{\lambda s_e}$ satisfies PT property

$$
\begin{array}{ll}
\text{(Sigma)} & \langle R \cup \{(a_{A_1}^{\Gamma_1}\ \sigma^i\, b_{A_2}^{\Gamma_2})_{A_3}^{\Gamma_3}\}, E\rangle \rightarrow \\
& \qquad \langle R, E \cup \{A_1{=}A_3, \Gamma_1{=}A'_1.\cdots.A'_{i-1}.A_2.\Gamma_2, \Gamma_3{=}A'_1.\cdots.A'_{i-1}.\Gamma_2\}\rangle, \\
\text{(Phi)} & \langle R \cup \{(\varphi_k^i\ a_{A_1}^{\Gamma_1})_{A_2}^{\Gamma_2}\}, E\rangle \rightarrow \\
& \qquad \langle R, E \cup \{A_1 = A_2, \Gamma_2 = A'_1.\cdots.A'_{k+i-1}.\Gamma', \Gamma_1 = A'_1.\cdots.A'_k.\Gamma'\}\rangle,
\end{array}
$$

The System $TA_{\lambda\sigma}$[ACCL91]

Syntax

Terms $a ::= \underline{1} \mid (a\ a) \mid \lambda.a \mid a[s]$ Substitution $s ::= id \mid \uparrow \mid a.s \mid s \circ s$

Typing rules

Terms

$$A.\Gamma \vdash \underline{1} : A\ (var) \qquad \frac{A.\Gamma \vdash b : B}{\Gamma \vdash \lambda.b : A \rightarrow B}\ (lambda)$$

$$\frac{\Gamma \vdash a : A \rightarrow B \quad \Gamma \vdash b : A}{\Gamma \vdash (a\ b) : B}\ (app) \qquad \frac{\Gamma \vdash s \rhd \Gamma' \quad \Gamma' \vdash a : A}{\Gamma \vdash a[s] : A}\ (clos)$$

Substitutions

$$\Gamma \vdash id \rhd \Gamma\ (id) \qquad A.\Gamma \vdash \uparrow \rhd \Gamma\ (shift)$$

$$\frac{\Gamma \vdash a : A \quad \Gamma \vdash s \rhd \Gamma'}{\Gamma \vdash a.s \rhd A.\Gamma'}\ (cons) \qquad \frac{\Gamma \vdash s'' \rhd \Gamma'' \quad \Gamma'' \vdash s' \rhd \Gamma'}{\Gamma \vdash s' \circ s'' \rhd \Gamma'}\ (comp)$$

PT in $TA_{\lambda\sigma}$

Lemma (Weakening)

If $\Gamma \vdash a : B$, then $\Gamma.A \vdash a : B$ for any type A. Particularly, if $\Gamma \vdash s \rhd \Gamma'$ then $\Gamma.A \vdash s \rhd \Gamma'.A$.

PT in $TA_{\lambda\sigma}$

Lemma (Weakening)

If $\Gamma \vdash a : B$, then $\Gamma.A \vdash a : B$ for any type A. Particularly, if $\Gamma \vdash s \triangleright \Gamma'$ then $\Gamma.A \vdash s \triangleright \Gamma'.A$.

Definition (PT in $TA_{\lambda\sigma}$)

Let $\tau = (\Gamma, \mathbb{T})$, where $\mathrm{TA}_{\lambda\sigma} \blacktriangleright a : \tau$.

PT in $TA_{\lambda\sigma}$

Lemma (Weakening)

If $\Gamma \vdash a : B$, then $\Gamma.A \vdash a : B$ for any type A. Particularly, if $\Gamma \vdash s \triangleright \Gamma'$ then $\Gamma.A \vdash s \triangleright \Gamma'.A$.

Definition (PT in $TA_{\lambda\sigma}$)

Let $\tau = (\Gamma, \mathbb{T})$, where $\mathrm{TA}_{\lambda\sigma} \blacktriangleright a : \tau.\tau$ is principal typing of a iff for any typing $\tau' = (\Gamma', \mathbb{T}')$ where $\mathrm{TA}_{\lambda\sigma} \blacktriangleright a : \tau'$, then exists some type substitution s such that $s(\Gamma) = \Gamma'_{\leq|\Gamma|}$ and: if $\mathbb{T}$ is a type then $s(\mathbb{T}) = \mathbb{T}'$ otherwise $s(\mathbb{T}) = \mathbb{T}'_{\leq|\mathbb{T}|}$.

PT for $TA_{\lambda\sigma}$

Theorem (Correspondence for $TA_{\lambda\sigma}$)

A typing τ is PT of b in $TA_{\lambda\sigma}$ iff τ is principal of b according to Wells' PT definition

PT for $TA_{\lambda\sigma}$

Theorem (Correspondence for $TA_{\lambda\sigma}$)

A typing τ is PT of b in $TA_{\lambda\sigma}$ iff τ is principal of b according to Wells' PT definition

Theorem (PT for $TA_{\lambda\sigma}$)

$TA_{\lambda\sigma}$ satisfies PT property

Type Inference for $TA_{\lambda\sigma}$ [Bo95]

$$
\begin{array}{lll}
(\mathrm{Var}) & \langle R \cup \{\underline{1}_A^{\Gamma}\}, E\rangle & \rightarrow \langle R, E \cup \{\Gamma = A.\Gamma'\}\rangle \\
(\mathrm{Lambda}) & \langle R \cup \{(\lambda.a_{A_1}^{\Gamma_1})_{A_2}^{\Gamma_2}\}, E\rangle & \rightarrow \langle R, E \cup \{A_2 = A^* \rightarrow A_1, \Gamma_1 = A^*.\Gamma_2\}\rangle \\
(\mathrm{App}) & \langle R \cup \{(a_{A_1}^{\Gamma_1}\ b_{A_2}^{\Gamma_2})_{A_3}^{\Gamma_3}\}, E\rangle & \rightarrow \langle R, E \cup \{\Gamma_1 = \Gamma_2, \Gamma_2 = \Gamma_3, A_1 = A_2 \rightarrow A_3\}\rangle \\
(\mathrm{Clos}) & \langle R \cup \{(a_{A_1}^{\Gamma_1}[s_{\Gamma_3}^{\Gamma_2}])_{A_2}^{\Gamma_4}\}, E\rangle & \rightarrow \langle R, E \cup \{\Gamma_1 = \Gamma_3, \Gamma_2 = \Gamma_4, A_1 = A_2\}\rangle \\
(\mathrm{Id}) & \langle R \cup \{id_{\Gamma_2}^{\Gamma_1}\}, E\rangle & \rightarrow \langle R, E \cup \{\Gamma_1 = \Gamma_2\}\rangle \\
(\mathrm{Shift}) & \langle R \cup \{\uparrow_{\Gamma_2}^{\Gamma_1}\}, E\rangle & \rightarrow \langle R, E \cup \{\Gamma_1 = A'.\Gamma_2\}\rangle \\
(\mathrm{Cons}) & \langle R \cup \{(a_{A_1}^{\Gamma_1}.s_{\Gamma_3}^{\Gamma_2})_{\Gamma_5}^{\Gamma_4}\}, E\rangle & \rightarrow \langle R, E \cup \{\Gamma_1 = \Gamma_2, \Gamma_2 = \Gamma_4, \Gamma_5 = A_1.\Gamma_3\}\rangle \\
(\mathrm{Comp}) & \langle R \cup \{(s_{\Gamma_2}^{\Gamma_1} \circ t_{\Gamma_4}^{\Gamma_3})_{\Gamma_6}^{\Gamma_5}\}, E\rangle & \rightarrow \langle R, E \cup \{\Gamma_1 = \Gamma_4, \Gamma_2 = \Gamma_6, \Gamma_3 = \Gamma_5\}\rangle
\end{array}
$$

Conclusion

- PT is not a trivial property

Conclusion

- PT is not a trivial property
- A definition of PT for λ-calculus in de Bruijn notation was proposed and proved correct

Conclusion

- PT is not a trivial property
- A definition of PT for λ-calculus in de Bruijn notation was proposed and proved correct
- The PT property for $TA_{\lambda dB}$ was proved

Conclusion

- PT is not a trivial property
- A definition of PT for λ-calculus in de Bruijn notation was proposed and proved correct
- The PT property for $TA_{\lambda dB}$ was proved
- A definition of PT for λs_e and $\lambda\sigma$ were proposed and proved correct

Conclusion

- PT is not a trivial property
- A definition of PT for λ-calculus in de Bruijn notation was proposed and proved correct
- The PT property for $TA_{\lambda dB}$ was proved
- A definition of PT for λs_e and $\lambda\sigma$ were proposed and proved correct
- The PT property were proved using a type inference algorithm for λs_e [AyMu2000] and $\lambda\sigma$ [Bo95].

Further Work

- Verify if the PT property holds for the corresponding systems with open terms.

Further Work

- Verify if the PT property holds for the corresponding systems with open terms.
- Research $\lambda\sigma$ and λs_e with intersection types and study PT for these new systems

Further Work

- Verify if the PT property holds for the corresponding systems with open terms.
- Research $\lambda\sigma$ and λs_e with intersection types and study PT for these new systems
- Type unification + substitution calls for *expansion variables*

M. Abadi, L. Cardelli, P.-L. Curien, and J.-J. Lévy.
Explicit Substitutions.
Journal of Functional Programming, 1(4):375–416, 1991. Cambridge University Press.

M. Ayala-Rincón and C. Muñoz.
Explicit Substitutions and All That.
Revista Colombiana de Computación, 1(1):47–71, 2000.

P. Borovanský.
Implementation of Higher-Order Unification Based on Calculus of Explicit Substitutions.
LNCS: Proceedings of the 22nd Seminar on Current Trends in Theory and Practice of Informatics (SOFSEM'95), 1012:363–368, 1995. Springer Verlag.

N.G. de Bruijn.
Lambda-Calculus Notation with Nameless Dummies, a Tool for Automatic Formula Manipulation, with Application to the Church-Rosser Theorem.
Indagationes Mathematicae, 34(5):381–392, 1972.

L. Damas and R. Milner.
Principal Type-Schemes for Functional Programs.
ACM Symposium on Principles of Programming Languages (POPL'82), 207–212, 1982. ACM Press.

J. R. Hindley.
Basic Simple Type Theory.
Cambridge Tracts in Theoretical Computer Science, 42, 1997. Cambridge University Press.

T. Jim.
What are principal typings and what are they good for?
ACM Symposium on Principles of Programming Languages (POPL'96), 42–53, 1996. ACM Press.

A.J. Kfoury and J.B. Wells
Principality and type inference for intersection types using expansion variables,
Theoretical Computer Science, 311(1-3):1–70, 2004. Elsevier.

F. Kamareddine and A. Ríos.
Extending a λ-calculus with explicit substitution which preserves strong normalisation into a confluent calculus on open terms,
Journal of Functional Programming, 7:395–420, 1997. Cambridge University Press.

J.B. Wells
The essence of principal typings,
LNCS: Proceedings of the 29th International Colloquium on Automata, Languages and Programming, 2380:913–925, 2002. Springer-Verlag.