

Tintin and the lost rewriting paradise

Jean-Pierre Jouannaud
INRIA-LIAMA and Tsinghua University
Beijing, China

November 22, 2008

Outline

- 1 Tintin
- 2 Encounter
- 3 Grenoble
- 4 Orsay
- 5 Cachan

Hergé
is the writer he praises most.

Tintin
is the character he feels closest.

Like Tintin,
Hubert
is chasing the truth everywhere, every day.

Like Tintin,
Hubert
is seeking both his personal truth and the universal truth.

Like Tintin,
his quest will never terminate.

PC Chair: Jörg Siekmann

one submission by Hubert Comon:

Sufficient completeness, term rewriting and anti-unification

H+1 references as of today.

The story

- The submission was very badly written
- The title was badly chosen, but
- It was fiercely attacking an important problem
- It contained two fundamental new ideas:
constraints and term languages
- I convinced the committee to select the paper
- I promised helping him reshape the paper and prepare the presentation
- We had a long preparatory discussion the first day of the conference
- The presentation was very well received

The lost rewriting paradise

Example

$$\mathcal{C} = \{0, 1, s\}$$

$$\mathcal{D} = \{+\}$$

$$\mathcal{R}_{\mathcal{C}} = \{ss(x) \rightarrow x, s(0) \rightarrow 1, s(1) \rightarrow 0\}$$

$$\mathcal{R}_{\mathcal{D}} = \{0 + 1 \rightarrow 0, 1 + 0 \rightarrow 1, x + x \rightarrow 0\}$$

Assumption: $\mathcal{R}_{\mathcal{C}} \cup \mathcal{R}_{\mathcal{D}}$ is convergent

Sufficient completeness: is + total on ground terms ?

[Guttag, Horning]:

Every ground term is equivalent to a constructor term.

[Jouannaud, Kounalis]:

The (irreducible) term $x + y$ is ground reducible.

[Comon]: The language defined by the 1st-order formula

$\exists xy.z = x + y \ \& \ z \neq 0 + 1 \ \& \ z \neq 1 + 0 \ \& \ \forall w.z \neq w + w$
interpreted over ground terms in normal form is empty.

All of a sudden, first-order logic and formal language theory
popped up: our good, old, abstract rewriting paradise was lost

The lost rewriting paradise

Example

$$\mathcal{C} = \{0, 1, s\}$$

$$\mathcal{D} = \{+\}$$

$$\mathcal{R}_{\mathcal{C}} = \{ss(x) \rightarrow x, s(0) \rightarrow 1, s(1) \rightarrow 0\}$$

$$\mathcal{R}_{\mathcal{D}} = \{0 + 1 \rightarrow 0, 1 + 0 \rightarrow 1, x + x \rightarrow 0\}$$

Assumption: $\mathcal{R}_{\mathcal{C}} \cup \mathcal{R}_{\mathcal{D}}$ is convergent

Sufficient completeness: is + total on ground terms ?

[Guttag, Horning]:

Every ground term is equivalent to a constructor term.

[Jouannaud, Kounalis]:

The (irreducible) term $x + y$ is ground reducible.

[Comon]: The language defined by the 1st-order formula

$\exists xy.z = x + y \ \& \ z \neq 0 + 1 \ \& \ z \neq 1 + 0 \ \& \ \forall w.z \neq w + w$
interpreted over ground terms in normal form is empty.

All of a sudden, first-order logic and formal language theory
popped up: our good, old, abstract rewriting paradise was lost

The lost rewriting paradise

Example

$$\begin{aligned}\mathcal{C} &= \{0, 1, s\} \\ \mathcal{D} &= \{+\} \\ \mathcal{R}_{\mathcal{C}} &= \{ss(x) \rightarrow x, s(0) \rightarrow 1, s(1) \rightarrow 0\} \\ \mathcal{R}_{\mathcal{D}} &= \{0 + 1 \rightarrow 0, 1 + 0 \rightarrow 1, x + x \rightarrow 0\}\end{aligned}$$

Assumption: $\mathcal{R}_{\mathcal{C}} \cup \mathcal{R}_{\mathcal{D}}$ is convergent

Sufficient completeness: is + total on ground terms ?

[Guttag, Horning]:

Every ground term is equivalent to a constructor term.

[Jouannaud, Kounalis]:

The (irreducible) term $x + y$ is ground reducible.

[Comon]: The language defined by the 1st-order formula
 $\exists xy.z = x + y \ \& \ z \neq 0 + 1 \ \& \ z \neq 1 + 0 \ \& \ \forall w.z \neq w + w$
interpreted over ground terms in normal form is empty.

All of a sudden, first-order logic and formal language theory
popped up: our good, old, abstract rewriting paradise was lost

The lost rewriting paradise

Example

$$\begin{aligned}\mathcal{C} &= \{0, 1, s\} \\ \mathcal{D} &= \{+\} \\ \mathcal{R}_{\mathcal{C}} &= \{ss(x) \rightarrow x, s(0) \rightarrow 1, s(1) \rightarrow 0\} \\ \mathcal{R}_{\mathcal{D}} &= \{0 + 1 \rightarrow 0, 1 + 0 \rightarrow 1, x + x \rightarrow 0\}\end{aligned}$$

Assumption: $\mathcal{R}_{\mathcal{C}} \cup \mathcal{R}_{\mathcal{D}}$ is convergent

Sufficient completeness: is + total on ground terms ?

[Guttag, Horning]:

Every ground term is equivalent to a constructor term.

[Jouannaud, Kounalis]:

The (irreducible) term $x + y$ is ground reducible.

[Comon]: The language defined by the 1st-order formula
 $\exists xy.z = x + y \ \& \ z \neq 0 + 1 \ \& \ z \neq 1 + 0 \ \& \ \forall w.z \neq w + w$
interpreted over ground terms in normal form is empty.

All of a sudden, first-order logic and formal language theory
popped up: our good, old, abstract rewriting paradise was lost

The lost rewriting paradise

Example

$$\mathcal{C} = \{0, 1, s\}$$

$$\mathcal{D} = \{+\}$$

$$\mathcal{R}_{\mathcal{C}} = \{ss(x) \rightarrow x, s(0) \rightarrow 1, s(1) \rightarrow 0\}$$

$$\mathcal{R}_{\mathcal{D}} = \{0 + 1 \rightarrow 0, 1 + 0 \rightarrow 1, x + x \rightarrow 0\}$$

Assumption: $\mathcal{R}_{\mathcal{C}} \cup \mathcal{R}_{\mathcal{D}}$ is convergent

Sufficient completeness: is + total on ground terms ?

[Guttag, Horning]:

Every ground term is equivalent to a constructor term.

[Jouannaud, Kounalis]:

The (irreducible) term $x + y$ is ground reducible.

[Comon]: The language defined by the 1st-order formula

$\exists xy.z = x + y \ \& \ z \neq 0 + 1 \ \& \ z \neq 1 + 0 \ \& \ \forall w.z \neq w + w$
interpreted over ground terms in normal form is empty.

All of a sudden, first-order logic and formal language theory
popped up: our good, old, abstract rewriting paradise was lost

Grenoble: building a friendship

Our recipee:

- Hard work
- Heavy smoking on Hubert's side
- Skying
- Climbing
- Good food
- and good bottles from my side ...

Our recipee:

- **Hard work**
- Heavy smoking on Hubert's side
- Skying
- Climbing
- Good food
- and good bottles from my side ...

Our recipee:

- Hard work
- Heavy smoking on Hubert's side
- Skying
- Climbing
- Good food
- and good bottles from my side ...

Grenoble: building a friendship

Our recipee:

- Hard work
- Heavy smoking on Hubert's side
- Skying
- Climbing
- Good food
- and good bottles from my side ...

Grenoble: building a friendship

Our recipe:

- Hard work
- Heavy smoking on Hubert's side
- Skying
- Climbing
- Good food
- and good bottles from my side ...

Grenoble: building a friendship

Our recipee:

- Hard work
- Heavy smoking on Hubert's side
- Skying
- Climbing
- Good food
- and good bottles from my side ...

Our recipee:

- Hard work
- Heavy smoking on Hubert's side
- Skying
- Climbing
- Good food
- and good bottles from my side ...

The blossoming period

Results of his PhD thesis:

- Result 1 [Malcev]: the first-order theory of trees is decidable.
- Result 2 [Malcev]: the first-order theory of trees quotiented by permutative axioms is decidable.
- Corollary 1: these theories are recursively axiomatizable.
- Result 3: Normal form languages are not recognizable, but correspond to a class of grammars for which emptiness is decidable.
- Corollary 2: ground reducibility is decidable for arbitrary \mathcal{R} .
- Result 4: lots of open problems raised, among which:
 - the decidability of elimination of negation
 - the systematic study of the class of normal form languages

The blossoming period

Results of his PhD thesis:

- **Result 1 [Malcev]:** the first-order theory of trees is decidable.
- **Result 2 [Malcev]:** the first-order theory of trees quotiented by permutative axioms is decidable.
- **Corollary 1:** these theories are recursively axiomatizable.
- **Result 3:** Normal form languages are not recognizable, but correspond to a class of grammars for which emptiness is decidable.
- **Corollary 2:** ground reducibility is decidable for arbitrary \mathcal{R} .
- **Result 4:** lots of open problems raised, among which:
 - the decidability of elimination of negation
 - the systematic study of the class of normal form languages

The blossoming period

Results of his PhD thesis:

- **Result 1 [Malcev]:** the first-order theory of trees is decidable.
- **Result 2 [Malcev]:** the first-order theory of trees quotiented by permutative axioms is decidable.
- **Corollary 1:** these theories are recursively axiomatizable.
- **Result 3:** Normal form languages are not recognizable, but correspond to a class of grammars for which emptiness is decidable.
- **Corollary 2:** ground reducibility is decidable for arbitrary \mathcal{R} .
- **Result 4:** lots of open problems raised, among which:
 - the decidability of elimination of negation
 - the systematic study of the class of normal form languages

The blossoming period

Results of his PhD thesis:

- **Result 1 [Malcev]:** the first-order theory of trees is decidable.
- **Result 2 [Malcev]:** the first-order theory of trees quotiented by permutative axioms is decidable.
- **Corollary 1:** these theories are recursively axiomatizable.
- **Result 3:** Normal form languages are not recognizable, but correspond to a class of grammars for which emptiness is decidable.
- **Corollary 2:** ground reducibility is decidable for arbitrary \mathcal{R} .
- **Result 4:** lots of open problems raised, among which:
 - the decidability of elimination of negation
 - the systematic study of the class of normal form languages

The blossoming period

Results of his PhD thesis:

- **Result 1 [Malcev]:** the first-order theory of trees is decidable.
- **Result 2 [Malcev]:** the first-order theory of trees quotiented by permutative axioms is decidable.
- **Corollary 1:** these theories are recursively axiomatizable.
- **Result 3:** Normal form languages are not recognizable, but correspond to a class of grammars for which emptiness is decidable.
- **Corollary 2:** ground reducibility is decidable for arbitrary \mathcal{R} .
- **Result 4:** lots of open problems raised, among which:
 - the decidability of elimination of negation
 - the systematic study of the class of normal form languages

The blossoming period

Results of his PhD thesis:

- **Result 1 [Malcev]:** the first-order theory of trees is decidable.
- **Result 2 [Malcev]:** the first-order theory of trees quotiented by permutative axioms is decidable.
- **Corollary 1:** these theories are recursively axiomatizable.
- **Result 3:** Normal form languages are not recognizable, but correspond to a class of grammars for which emptiness is decidable.
- **Corollary 2:** ground reducibility is decidable for arbitrary \mathcal{R} .
- **Result 4:** lots of open problems raised, among which:
 - the decidability of elimination of negation
 - the systematic study of the class of normal form languages

The blossoming period

Results of his PhD thesis:

- **Result 1 [Malcev]:** the first-order theory of trees is decidable.
- **Result 2 [Malcev]:** the first-order theory of trees quotiented by permutative axioms is decidable.
- **Corollary 1:** these theories are recursively axiomatizable.
- **Result 3:** Normal form languages are not recognizable, but correspond to a class of grammars for which emptiness is decidable.
- **Corollary 2:** ground reducibility is decidable for arbitrary \mathcal{R} .
- **Result 4:** lots of open problems raised, among which:
 - the decidability of elimination of negation
 - the systematic study of the class of normal form languages

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
 - Complement problems and negation elimination
 - Presburger constraints
 - Terms with integer exponents
 - Inductionless induction
 - Ordering constraints
 - Confluence of rewrite rules with membership constraints
 - Sequentiality
 - Higher-order matching
 - Ground reducibility is exptime complete
 - Ground confluence is polynomial
 - and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Orsay : the harvest

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

- Types and Computer Algebra
- Shallow theories
- Complement problems and negation elimination
- Presburger constraints
- Terms with integer exponents
- Inductionless induction
- Ordering constraints
- Confluence of rewrite rules with membership constraints
- Sequentiality
- Higher-order matching
- Ground reducibility is exptime complete
- Ground confluence is polynomial
- and many more

Inductive proofs by consistency

- Given a set of equations E and a set of conjectures C , add C to E and try to derive an inconsistency.
- [Musser]: use a completely defined equality predicate
- [Huet and Hullot]: constructor discipline
- [Jouannaud and Kounalis]: inductive reducibility
- [Fribourg]: linear strategy
- [Bachmair]: unfailing completion
- Comon]: model-theoretic characterisation of proof by consistency which captures all above cases and yields new ones.

Inductive proofs by consistency

- Given a set of equations E and a set of conjectures C , add C to E and try to derive an inconsistency.
- **[Musser]**: use a completely defined equality predicate
- [Huet and Hullot]: constructor discipline
- [Jouannaud and Kounalis]: inductive reducibility
- [Fribourg]: linear strategy
- [Bachmair]: unfailing completion
- Comon]: model-theoretic characterisation of proof by consistency which captures all above cases and yields new ones.

Inductive proofs by consistency

- Given a set of equations E and a set of conjectures C , add C to E and try to derive an inconsistency.
- **[Musser]**: use a completely defined equality predicate
- **[Huet and Hullot]**: constructor discipline
- **[Jouannaud and Kounalis]**: inductive reducibility
- **[Fribourg]**: linear strategy
- **[Bachmair]**: unfailing completion
- **Comon**: model-theoretic characterisation of proof by consistency which captures all above cases and yields new ones.

Inductive proofs by consistency

- Given a set of equations E and a set of conjectures C , add C to E and try to derive an inconsistency.
- [Musser]: use a completely defined equality predicate
- [Huet and Hullot]: constructor discipline
- [Jouannaud and Kounalis]: inductive reducibility
- [Fribourg]: linear strategy
- [Bachmair]: unfailing completion
- Comon]: model-theoretic characterisation of proof by consistency which captures all above cases and yields new ones.

Inductive proofs by consistency

- Given a set of equations E and a set of conjectures C , add C to E and try to derive an inconsistency.
- [Musser]: use a completely defined equality predicate
- [Huet and Hullot]: constructor discipline
- [Jouannaud and Kounalis]: inductive reducibility
- [Fribourg]: linear strategy
- [Bachmair]: unfailing completion
- Comon]: model-theoretic characterisation of proof by consistency which captures all above cases and yields new ones.

Inductive proofs by consistency

- Given a set of equations E and a set of conjectures C , add C to E and try to derive an inconsistency.
- [Musser]: use a completely defined equality predicate
- [Huet and Hullot]: constructor discipline
- [Jouannaud and Kounalis]: inductive reducibility
- [Fribourg]: linear strategy
- [Bachmair]: unfailing completion
- Comon]: model-theoretic characterisation of proof by consistency which captures all above cases and yields new ones.

Inductive proofs by consistency

- Given a set of equations E and a set of conjectures C , add C to E and try to derive an inconsistency.
- [Musser]: use a completely defined equality predicate
- [Huet and Hullot]: constructor discipline
- [Jouannaud and Kounalis]: inductive reducibility
- [Fribourg]: linear strategy
- [Bachmair]: unfailing completion
- Comon]: model-theoretic characterisation of proof by consistency which captures all above cases and yields new ones.

Symbolic ordering constraints LICS'

- Lexicographic path ordering for ground terms generated by a well-founded order $>_{\mathcal{F}}$ on the signature \mathcal{F} :

$$s = f(\bar{s}) \succ g(\bar{t}) = t \quad \text{iff}$$

subterm case: $\exists u \in \bar{s}, u \succeq t$

precedence case: $f >_{\mathcal{F}} g$ and $\forall v \in \bar{t}, s \succ v$

lexicographic case: $f = g$ and $\forall v \in \bar{t} s \succ v$ and $\bar{s} \succ_{lex} \bar{t}$

- **Huet:** how to define a *stable* order \succ on open terms:
 $s \succ t$ implies $s\sigma \succ t\sigma$ for all σ ?
- **Dershowitz:**
variables are incomparable with everything else
- **Comon:**
 $s \succ t$ iff the formula $t \not\prec s$ is unsatisfiable on ground terms,
which is (surprisingly) decidable.
Comon and Treinen:
The first-order theory of lpo is undecidable

Symbolic ordering constraints LICS'

- Lexicographic path ordering for ground terms generated by a well-founded order $>_{\mathcal{F}}$ on the signature \mathcal{F} :

$$s = f(\bar{s}) \succ g(\bar{t}) = t \quad \text{iff}$$

subterm case: $\exists u \in \bar{s}, u \succeq t$

precedence case: $f >_{\mathcal{F}} g$ and $\forall v \in \bar{t}, s \succ v$

lexicographic case: $f = g$ and $\forall v \in \bar{t} s \succ v$ and $\bar{s} \succ_{lex} \bar{t}$

- **Huet:** how to define a *stable* order \succ on open terms:
 $s \succ t$ implies $s\sigma \succ t\sigma$ for all σ ?

- **Dershowitz:**

variables are incomparable with everything else

- **Comon:**

$s \succ t$ iff the formula $t \not\prec s$ is unsatisfiable on ground terms, which is (surprisingly) decidable.

Comon and Treinen:

The first-order theory of lpo is undecidable

Symbolic ordering constraints LICS'

- Lexicographic path ordering for ground terms generated by a well-founded order $>_{\mathcal{F}}$ on the signature \mathcal{F} :

$$s = f(\bar{s}) \succ g(\bar{t}) = t \quad \text{iff}$$

subterm case: $\exists u \in \bar{s}, u \succeq t$

precedence case: $f >_{\mathcal{F}} g$ and $\forall v \in \bar{t}, s \succ v$

lexicographic case: $f = g$ and $\forall v \in \bar{t} s \succ v$ and $\bar{s} \succ_{lex} \bar{t}$

- **Huet:** how to define a *stable* order \succ on open terms:
 $s \succ t$ implies $s\sigma \succ t\sigma$ for all σ ?

- **Dershowitz:**
variables are incomparable with everything else

- **Comon:**
 $s \succ t$ iff the formula $t \not\prec s$ is unsatisfiable on ground terms,
which is (surprisingly) decidable.

Comon and Treinen:

The first-order theory of lpo is undecidable

Symbolic ordering constraints LICS'

- Lexicographic path ordering for ground terms generated by a well-founded order $>_{\mathcal{F}}$ on the signature \mathcal{F} :

$$s = f(\bar{s}) \succ g(\bar{t}) = t \quad \text{iff}$$

subterm case: $\exists u \in \bar{s}, u \succeq t$

precedence case: $f >_{\mathcal{F}} g$ and $\forall v \in \bar{t}, s \succ v$

lexicographic case: $f = g$ and $\forall v \in \bar{t} s \succ v$ and $\bar{s} \succ_{lex} \bar{t}$

- Huet:** how to define a *stable* order \succ on open terms:
 $s \succ t$ implies $s\sigma \succ t\sigma$ for all σ ?
- Dershowitz:**
variables are incomparable with everything else
- Comon:**
 $s \succ t$ iff the formula $t \not\prec s$ is unsatisfiable on ground terms,
which is (surprisingly) decidable.
Comon and Treinen:
The first-order theory of lpo is undecidable

Order-sorted rewriting in OBJ

- `obj NAT+`
 `sorts Zero, Pos, Nat`
 `subsorts Zero < Nat, Pos < Nat`
 `cons 0: Zero`
 `cons succ : Nat -> Pos`
 `cons pred : Pos -> Nat`
 `var x: Nat, y: Pos`
 `eq pred(succ(x)) = x`
 `eq succ(pred(y)) = y`
 `end obj`
- **Comon:** Order-sorted signatures are bottom-up tree automata
- **Comon:** Normal forms are “recognizable” and define the quotient algebra if the rules are Church-Rosser
- **Puzzle:** Rewrite rules without critical pairs may be non-confluent in case rewriting is sort-increasing
- **Comon:** The hidden critical pairs can be computed by a decidable fragment of second-order unification

Order-sorted rewriting in OBJ

- obj NAT+

```
sorts Zero, Pos, Nat
subsorts Zero < Nat, Pos < Nat
cons 0: Zero
cons succ : Nat -> Pos
cons pred : Pos -> Nat
var x: Nat, y: Pos
eq pred(succ(x)) = x
eq succ(pred(y)) = y
end obj
```

- **Comon:** Order-sorted signatures are bottom-up tree automata

- **Comon:** Normal forms are “recognizable” and define the quotient algebra if the rules are Church-Rosser
- **Puzzle:** Rewrite rules without critical pairs may be non-confluent in case rewriting is sort-increasing
- **Comon:** The hidden critical pairs can be computed by a decidable fragment of second-order unification

Order-sorted rewriting in OBJ

- obj NAT+
sorts Zero, Pos, Nat
subsorts Zero < Nat, Pos < Nat
cons 0: Zero
cons succ : Nat -> Pos
cons pred : Pos -> Nat
var x: Nat, y: Pos
eq pred(succ(x)) = x
eq succ(pred(y)) = y
end obj
- **Comon:** Order-sorted signatures are bottom-up tree automata
- **Comon:** Normal forms are “recognizable” and define the quotient algebra if the rules are Church-Rosser
- **Puzzle:** Rewrite rules without critical pairs may be non-confluent in case rewriting is sort-increasing
- **Comon:** The hidden critical pairs can be computed by a decidable fragment of second-order unification

Order-sorted rewriting in OBJ

- `obj NAT+`
`sorts Zero, Pos, Nat`
`subsorts Zero < Nat, Pos < Nat`
`cons 0: Zero`
`cons succ : Nat -> Pos`
`cons pred : Pos -> Nat`
`var x: Nat, y: Pos`
`eq pred(succ(x)) = x`
`eq succ(pred(y)) = y`
`end obj`
- **Comon:** Order-sorted signatures are bottom-up tree automata
- **Comon:** Normal forms are “recognizable” and define the quotient algebra if the rules are Church-Rosser
- **Puzzle:** Rewrite rules without critical pairs may be non-confluent in case rewriting is sort-increasing
- **Comon:** The hidden critical pairs can be computed by a decidable fragment of second-order unification

Order-sorted rewriting in OBJ

- `obj NAT+`
`sorts Zero, Pos, Nat`
`subsorts Zero < Nat, Pos < Nat`
`cons 0: Zero`
`cons succ : Nat -> Pos`
`cons pred : Pos -> Nat`
`var x: Nat, y: Pos`
`eq pred(succ(x)) = x`
`eq succ(pred(y)) = y`
`end obj`
- **Comon:** Order-sorted signatures are bottom-up tree automata
- **Comon:** Normal forms are “recognizable” and define the quotient algebra if the rules are Church-Rosser
- **Puzzle:** Rewrite rules without critical pairs may be non-confluent in case rewriting is sort-increasing
- **Comon:** The hidden critical pairs can be computed by a decidable fragment of second-order unification

A specification of integers

Figure: A normal-form automaton for integers

Needed rewriting strategies

- The lambda-calculus is a rewrite system for which the normal form of a term, whenever it exists, can always be obtained by reducing the leftmost-innermost redex. Further, only **needed** redexes need be reduced.
- **Huet-Levy**: strongly sequential, orthogonal rewrite systems enjoy a similar property.
- **Klop,Middeldorp**: Simpler proof
- **Jouannaud,Sadfi**: Generalization to left-linear systems
- **Oyamaguchi**: Generalization to NV-sequentiality
- **Comon**: These problems can be uniformly formulated as a MSO formula, allowing to invoke Rabin's theorem
- **Comon**: Construction in EXPTIME of the automaton of needed redexes

Needed rewriting strategies

- The lambda-calculus is a rewrite system for which the normal form of a term, whenever it exists, can always be obtained by reducing the leftmost-innermost redex. Further, only **needed** redexes need be reduced.
- **Huet-Levy**: strongly sequential, orthogonal rewrite systems enjoy a similar property.
- **Klop,Middeldorp**: Simpler proof
- **Jouannaud,Sadfi**: Generalization to left-linear systems
- **Oyamaguchi**: Generalization to NV-sequentiality
- **Comon**: These problems can be uniformly formulated as a MSO formula, allowing to invoke Rabin's theorem
- **Comon**: Construction in EXPTIME of the automaton of needed redexes

Needed rewriting strategies

- The lambda-calculus is a rewrite system for which the normal form of a term, whenever it exists, can always be obtained by reducing the leftmost-innermost redex. Further, only **needed** redexes need be reduced.
- **Huet-Levy**: strongly sequential, orthogonal rewrite systems enjoy a similar property.
- **Klop,Middledorp**: Simpler proof
- **Jouannaud,Sadfi**: Generalization to left-linear systems
- **Oyamaguchi**: Generalization to NV-sequentiality
- **Comon**: These problems can be uniformly formulated as a MSO formula, allowing to invoke Rabin's theorem
- **Comon**: Construction in EXPTIME of the automaton of needed redexes

Needed rewriting strategies

- The lambda-calculus is a rewrite system for which the normal form of a term, whenever it exists, can always be obtained by reducing the leftmost-innermost redex. Further, only **needed** redexes need be reduced.
- **Huet-Levy**: strongly sequential, orthogonal rewrite systems enjoy a similar property.
- **Klop,Middledorp**: Simpler proof
- **Jouannaud,Sadfi**: Generalization to left-linear systems
- **Oyamaguchi**: Generalization to NV-sequentiality
- **Comon**: These problems can be uniformly formulated as a MSO formula, allowing to invoke Rabin's theorem
- **Comon**: Construction in EXPTIME of the automaton of needed redexes

Needed rewriting strategies

- The lambda-calculus is a rewrite system for which the normal form of a term, whenever it exists, can always be obtained by reducing the leftmost-innermost redex. Further, only **needed** redexes need be reduced.
- **Huet-Levy**: strongly sequential, orthogonal rewrite systems enjoy a similar property.
- **Klop,Middledorp**: Simpler proof
- **Jouannaud,Sadfi**: Generalization to left-linear systems
- **Oyamaguchi**: Generalization to NV-sequentiality
- **Comon**: These problems can be uniformly formulated as a MSO formula, allowing to invoke Rabin's theorem
- **Comon**: Construction in EXPTIME of the automaton of needed redexes

Needed rewriting strategies

- The lambda-calculus is a rewrite system for which the normal form of a term, whenever it exists, can always be obtained by reducing the leftmost-innermost redex. Further, only **needed** redexes need be reduced.
- **Huet-Levy**: strongly sequential, orthogonal rewrite systems enjoy a similar property.
- **Klop, Middeldorp**: Simpler proof
- **Jouannaud, Sadfi**: Generalization to left-linear systems
- **Oyamaguchi**: Generalization to NV-sequentiality
- **Comon**: These problems can be uniformly formulated as a MSO formula, allowing to invoke Rabin's theorem
- **Comon**: Construction in EXPTIME of the automaton of needed redexes

Needed rewriting strategies

- The lambda-calculus is a rewrite system for which the normal form of a term, whenever it exists, can always be obtained by reducing the leftmost-innermost redex. Further, only **needed** redexes need be reduced.
- **Huet-Levy**: strongly sequential, orthogonal rewrite systems enjoy a similar property.
- **Klop,Middledorp**: Simpler proof
- **Jouannaud,Sadfi**: Generalization to left-linear systems
- **Oyamaguchi**: Generalization to NV-sequentiality
- **Comon**: These problems can be uniformly formulated as a MSO formula, allowing to invoke Rabin's theorem
- **Comon**: Construction in EXPTIME of the automaton of needed redexes

Higher-order matching

- **Problem:**
solve functional equations $s = t$ over typed lambda-terms:
 t a simply typed ground lambda-term,
 s a simply typed lambda-term,
 σ , a simply typed substitution satisfying $s\sigma \longrightarrow_{\beta\eta} t$
- **folklore:** 1st-order case straightforward: ≤ 1 solution
- **Huet:** 2nd-order case follows from higher-order unification:
finitely many solutions
- **Dowek:** 3rd-order case decidable: ∞ -many solutions
- **Padovani:** 4th-order case decidable: ∞ -many solutions
- **Comon:** the set of solutions is recognizable up to order 4:
the bottom-up tree automaton has for states solutions of
finitely many problems of order 4-2.
- **Open problem:** What class of languages corresponds to a
problem of order $n \geq 5$?

Higher-order matching

- **Problem:**
solve functional equations $s = t$ over typed lambda-terms:
 t a simply typed ground lambda-term,
 s a simply typed lambda-term,
 σ , a simply typed substitution satisfying $s\sigma \longrightarrow_{\beta\eta} t$
- **folklore:** 1st-order case straightforward: ≤ 1 solution
- **Huet:** 2nd-order case follows from higher-order unification:
finitely many solutions
- **Dowek:** 3rd-order case decidable: ∞ -many solutions
- **Padovani:** 4th-order case decidable: ∞ -many solutions
- **Comon:** the set of solutions is recognizable up to order 4:
the bottom-up tree automaton has for states solutions of
finitely many problems of order 4-2.
- **Open problem:** What class of languages corresponds to a
problem of order $n \geq 5$?

Higher-order matching

- **Problem:**
solve functional equations $s = t$ over typed lambda-terms:
 t a simply typed ground lambda-term,
 s a simply typed lambda-term,
 σ , a simply typed substitution satisfying $s\sigma \longrightarrow_{\beta\eta} t$
- **folklore:** 1st-order case straightforward: ≤ 1 solution
- **Huet:** 2nd-order case follows from higher-order unification:
finitely many solutions
- **Dowek:** 3rd-order case decidable: ∞ -many solutions
- **Padovani:** 4th-order case decidable: ∞ -many solutions
- **Comon:** the set of solutions is recognizable up to order 4:
the bottom-up tree automaton has for states solutions of
finitely many problems of order 4-2.
- **Open problem:** What class of languages corresponds to a
problem of order $n \geq 5$?

Higher-order matching

- **Problem:**
solve functional equations $s = t$ over typed lambda-terms:
 t a simply typed ground lambda-term,
 s a simply typed lambda-term,
 σ , a simply typed substitution satisfying $s\sigma \longrightarrow_{\beta\eta} t$
- **folklore:** 1st-order case straightforward: ≤ 1 solution
- **Huet:** 2nd-order case follows from higher-order unification:
finitely many solutions
- **Dowek:** 3rd-order case decidable: ∞ -many solutions
- **Padovani:** 4th-order case decidable: ∞ -many solutions
- **Comon:** the set of solutions is recognizable up to order 4:
the bottom-up tree automaton has for states solutions of
finitely many problems of order 4-2.
- **Open problem:** What class of languages corresponds to a
problem of order $n \geq 5$?

Higher-order matching

- **Problem:**
solve functional equations $s = t$ over typed lambda-terms:
 t a simply typed ground lambda-term,
 s a simply typed lambda-term,
 σ , a simply typed substitution satisfying $s\sigma \longrightarrow_{\beta\eta} t$
- **folklore:** 1st-order case straightforward: ≤ 1 solution
- **Huet:** 2nd-order case follows from higher-order unification:
finitely many solutions
- **Dowek:** 3rd-order case decidable: ∞ -many solutions
- **Padovani:** 4th-order case decidable: ∞ -many solutions
- **Comon:** the set of solutions is recognizable up to order 4:
the bottom-up tree automaton has for states solutions of
finitely many problems of order 4-2.
- **Open problem:** What class of languages corresponds to a
problem of order $n \geq 5$?

Higher-order matching

- **Problem:**
solve functional equations $s = t$ over typed lambda-terms:
 t a simply typed ground lambda-term,
 s a simply typed lambda-term,
 σ , a simply typed substitution satisfying $s\sigma \longrightarrow_{\beta\eta} t$
- **folklore:** 1st-order case straightforward: ≤ 1 solution
- **Huet:** 2nd-order case follows from higher-order unification:
finitely many solutions
- **Dowek:** 3rd-order case decidable: ∞ -many solutions
- **Padovani:** 4th-order case decidable: ∞ -many solutions
- **Comon:** the set of solutions is recognizable up to order 4:
the bottom-up tree automaton has for states solutions of
finitely many problems of order 4-2.
- **Open problem:** What class of languages corresponds to a
problem of order $n \geq 5$?

Higher-order matching

- **Problem:**
solve functional equations $s = t$ over typed lambda-terms:
 t a simply typed ground lambda-term,
 s a simply typed lambda-term,
 σ , a simply typed substitution satisfying $s\sigma \longrightarrow_{\beta\eta} t$
- **folklore:** 1st-order case straightforward: ≤ 1 solution
- **Huet:** 2nd-order case follows from higher-order unification:
finitely many solutions
- **Dowek:** 3rd-order case decidable: ∞ -many solutions
- **Padovani:** 4th-order case decidable: ∞ -many solutions
- **Comon:** the set of solutions is recognizable up to order 4:
the bottom-up tree automaton has for states solutions of
finitely many problems of order 4-2.
- **Open problem:** What class of languages corresponds to a
problem of order $n \geq 5$?

- **TATA:**
An open-source electronic book about tree automata.
- Model checking more expressive logical formalisms:
Timed automata and the theory of real numbers
[CONCUR]
- Security: public key protocols without perfect encryption:
Intruder theories **[FOSSACS]**

Conclusion

- Hubert is a problem solver
 - He renews the way we look at problems
 - He is helped by his incredible mastering of technical tools
 - He is not only interested in decidability results, but also in their complexity, and in undecidability results as well
-
- His career has for sure many productive years ahead
 - if the authorities let him develop his ideas in peace

- Hubert is a problem solver
 - He renews the way we look at problems
 - He is helped by his incredible mastering of technical tools
 - He is not only interested in decidability results, but also in their complexity, and in undecidability results as well
-
- His career has for sure many productive years ahead
 - if the authorities let him develop his ideas in peace