

# Confluence Properties on Open Terms in the First-Order Theory of Rewriting

Franziska Rapp

joint work with

Aart Middeldorp

University of Innsbruck

IWC 2016, Obergurgl, September 8



# FORT

property



decision mode



TRS



yes | no | ?

# FORT

property



synthesis mode



TRS



no | ?

# FORT

property



TRS

$$\forall s \exists t (s \rightarrow^* t \wedge \neg \exists u (t \rightarrow u)) \\ \implies \exists v (s \twoheadrightarrow v \vee v \xrightarrow{\epsilon} t)$$



yes | no | ?

# FORT

property



TRS

$$\forall s \exists t (s \rightarrow^* t \wedge \neg \exists u (t \rightarrow u)) \\ \implies \exists v (s \twoheadrightarrow v \vee v \xrightarrow{\epsilon} t)$$



yes | no | ?

**FORT** is based on tree automata techniques (Dauchet and Tison, LICS 1990)

# Outline

- First-Order Theory of Rewriting
- Automation
- Properties on Open Terms
- Experiments
- Future Work

## First-Order Theory of Rewriting

- first-order logic over a language  $\mathcal{L}$  without function symbols

## First-Order Theory of Rewriting

- first-order logic over a language  $\mathcal{L}$  without function symbols
- $\mathcal{L}$  contains the following binary predicate symbols:

$\rightarrow$     $\rightarrow^+$     $\rightarrow^*$     $\rightarrow^!$     $\nrightarrow$     $\xrightarrow{\epsilon}$     $\xrightarrow{>\epsilon}$     $\leftrightarrow$     $\leftrightarrow^*$     $\downarrow$     $=$



## First-Order Theory of Rewriting

- first-order logic over a language  $\mathcal{L}$  without function symbols
- $\mathcal{L}$  contains the following binary predicate symbols:

$\rightarrow$     $\rightarrow^+$     $\rightarrow^*$     $\rightarrow^!$     $\nrightarrow$     $\xrightarrow{\epsilon}$     $\xrightarrow{>\epsilon}$     $\leftrightarrow$     $\leftrightarrow^*$     $\downarrow$     $=$

- models of  $\mathcal{L}$  are finite **left-linear right-ground** TRSs

## First-Order Theory of Rewriting

- first-order logic over a language  $\mathcal{L}$  without function symbols
- $\mathcal{L}$  contains the following binary predicate symbols:

$\rightarrow$     $\rightarrow^+$     $\rightarrow^*$     $\rightarrow^!$     $\nrightarrow$     $\xrightarrow{\epsilon}$     $\xrightarrow{>\epsilon}$     $\leftrightarrow$     $\leftrightarrow^*$     $\downarrow$     $=$

- models of  $\mathcal{L}$  are finite left-linear right-ground TRSs
- set of **ground terms** serves as domain for variables in formulas over  $\mathcal{L}$

## First-Order Theory of Rewriting

- first-order logic over a language  $\mathcal{L}$  without function symbols
- $\mathcal{L}$  contains the following binary predicate symbols:

$\rightarrow$     $\rightarrow^+$     $\rightarrow^*$     $\rightarrow^!$     $\nrightarrow$     $\xrightarrow{\epsilon}$     $\xrightarrow{>\epsilon}$     $\leftrightarrow$     $\leftrightarrow^*$     $\downarrow$     $=$

- models of  $\mathcal{L}$  are finite left-linear right-ground TRSs
- set of ground terms serves as domain for variables in formulas over  $\mathcal{L}$

## Derived Predicates

$$s \rightarrow^* t \iff s \rightarrow^+ t \vee s = t$$

$$s \leftrightarrow t \iff s \rightarrow t \vee t \rightarrow s$$

$$s \rightarrow^! t \iff s \rightarrow^* t \wedge \neg \exists u (t \rightarrow u)$$

$$s \downarrow t \iff \exists u (s \rightarrow^* u \wedge t \rightarrow^* u)$$

## Derived Predicates

$$s \rightarrow^* t \iff s \rightarrow^+ t \vee s = t$$

$$s \leftrightarrow t \iff s \rightarrow t \vee t \rightarrow s$$

$$s \rightarrow^! t \iff s \rightarrow^* t \wedge \neg \exists u (t \rightarrow u) \quad s \downarrow t \iff \exists u (s \rightarrow^* u \wedge t \rightarrow^* u)$$

$$\text{CR}(t) \iff \forall u \forall v (t \rightarrow^* u \wedge t \rightarrow^* v \implies u \downarrow v)$$

## Derived Predicates

$$s \rightarrow^* t \iff s \rightarrow^+ t \vee s = t$$

$$s \leftrightarrow t \iff s \rightarrow t \vee t \rightarrow s$$

$$s \rightarrow^! t \iff s \rightarrow^* t \wedge \neg \exists u (t \rightarrow u) \quad s \downarrow t \iff \exists u (s \rightarrow^* u \wedge t \rightarrow^* u)$$

$$\text{CR}(t) \iff \forall u \forall v (t \rightarrow^* u \wedge t \rightarrow v \implies u \downarrow v) \quad \text{CR} \iff \forall t \text{CR}(t)$$

## Derived Predicates

$$s \rightarrow^* t \iff s \rightarrow^+ t \vee s = t \qquad s \leftrightarrow t \iff s \rightarrow t \vee t \rightarrow s$$

$$s \rightarrow^! t \iff s \rightarrow^* t \wedge \neg \exists u (t \rightarrow u) \qquad s \downarrow t \iff \exists u (s \rightarrow^* u \wedge t \rightarrow^* u)$$

$$\text{CR}(t) \iff \forall u \forall v (t \rightarrow^* u \wedge t \rightarrow v \implies u \downarrow v) \qquad \text{CR} \iff \forall t \text{CR}(t)$$

$$\text{WCR}(t) \iff \forall u \forall v (t \rightarrow u \wedge t \rightarrow v \implies u \downarrow v) \qquad \text{WCR} \iff \forall t \text{WCR}(t)$$

$$\text{WN}(t) \iff \exists u (t \rightarrow^! u) \qquad \text{WN} \iff \forall t \text{WN}(t)$$

$$\text{UN}(t) \iff \forall u \forall v (t \rightarrow^! u \wedge t \rightarrow^! v \implies u = v) \qquad \text{UN} \iff \forall t \text{UN}(t)$$

$$\text{NFP}(t) \iff \forall u \forall v (t \rightarrow u \wedge t \rightarrow^! v \implies u \rightarrow^! v) \qquad \text{NFP} \iff \forall t \text{NFP}(t)$$

$$\text{NF}(t) \iff \neg \exists u (t \rightarrow u)$$

$$\text{UNC} \iff \forall t \forall u (t \leftrightarrow^* u \wedge \text{NF}(t) \wedge \text{NF}(u) \implies t = u)$$

# Outline

- First-Order Theory of Rewriting
- **Automation**
- Properties on Open Terms
- Experiments
- Future Work



## Translation

- **binary predicates** are  $RR_2$  relations and implemented via tree automata
  - ground tree transducers (**GTTs**) for  $\dashv\vdash$
  - **$RR_n$  automata** for all relations

## Translation

- binary predicates are  $RR_2$  relations and implemented via tree automata
  - ground tree transducers (GTTs) for  $\nrightarrow$
  - $RR_n$  automata for all relations
- **implications** and **universal quantifiers** are eliminated

$$\varphi \Rightarrow \psi \equiv \neg \varphi \vee \psi$$

$$\forall x \varphi \equiv \neg \exists x \neg \varphi$$

## Translation

- binary predicates are  $RR_2$  relations and implemented via tree automata
  - ground tree transducers (GTTs) for  $\nrightarrow$
  - $RR_n$  automata for all relations
- implications and universal quantifiers are eliminated

$$\varphi \Rightarrow \psi \equiv \neg \varphi \vee \psi$$

$$\forall x \varphi \equiv \neg \exists x \neg \varphi$$

- **negations** are pushed inside and double negations are eliminated

## Translation

- binary predicates are  $RR_2$  relations and implemented via tree automata
  - ground tree transducers (GTTs) for  $\nrightarrow$
  - $RR_n$  automata for all relations
- implications and universal quantifiers are eliminated

$$\varphi \Rightarrow \psi \equiv \neg \varphi \vee \psi$$

$$\forall x \varphi \equiv \neg \exists x \neg \varphi$$

- negations are pushed inside and double negations are eliminated
- remaining **propositional connectives** are implemented by corresponding closure operations on  $RR_n$  automata

## Translation

- binary predicates are  $RR_2$  relations and implemented via tree automata
  - ground tree transducers (GTTs) for  $\nrightarrow$
  - $RR_n$  automata for all relations
- implications and universal quantifiers are eliminated

$$\varphi \Rightarrow \psi \equiv \neg \varphi \vee \psi$$

$$\forall x \varphi \equiv \neg \exists x \neg \varphi$$

- negations are pushed inside and double negations are eliminated
- remaining propositional connectives are implemented by corresponding closure operations on  $RR_n$  automata
- **existential quantifiers** are implemented using projection

## Translation

- binary predicates are  $RR_2$  relations and implemented via tree automata
  - ground tree transducers (GTTs) for  $\nrightarrow$
  - $RR_n$  automata for all relations
- implications and universal quantifiers are eliminated

$$\varphi \Rightarrow \psi \equiv \neg \varphi \vee \psi \qquad \forall x \varphi \equiv \neg \exists x \neg \varphi$$

- negations are pushed inside and double negations are eliminated
- remaining propositional connectives are implemented by corresponding closure operations on  $RR_n$  automata
- existential quantifiers are implemented using projection

## Remark

formulas are **not** transformed into **prenex normal form**, since this increases the dimension of involved relations

# Outline

- First-Order Theory of Rewriting
- Automation
- Properties on Open Terms
- Experiments
- Future Work

## Definition

TRS is **confluent** if  $\forall s \forall t \forall u (s \rightarrow^* t \wedge s \rightarrow^* u \implies \exists v (t \rightarrow^* v \wedge u \rightarrow^* v))$



## Definition

TRS is confluent if  $\forall s \forall t \forall u (s \rightarrow^* t \wedge s \rightarrow^* u \implies \exists v (t \rightarrow^* v \wedge u \rightarrow^* v))$

## Remarks

- variables  $s, t, u, v$  range over all terms

## Definition

TRS is confluent if  $\forall s \forall t \forall u (s \rightarrow^* t \wedge s \rightarrow^* u \implies \exists v (t \rightarrow^* v \wedge u \rightarrow^* v))$

## Remarks

- variables  $s, t, u, v$  range over all terms
- FORT is based on tree automata techniques and (hence) variables range over **ground** terms

## Definition

TRS is confluent if  $\forall s \forall t \forall u (s \rightarrow^* t \wedge s \rightarrow^* u \implies \exists v (t \rightarrow^* v \wedge u \rightarrow^* v))$

## Remarks

- variables  $s, t, u, v$  range over all terms
- FORT is based on tree automata techniques and (hence) variables range over ground terms
- confluence  $\neq$  **ground-confluence**

## Definition

TRS is confluent if  $\forall s \forall t \forall u (s \rightarrow^* t \wedge s \rightarrow^* u \implies \exists v (t \rightarrow^* v \wedge u \rightarrow^* v))$

## Remarks

- variables  $s, t, u, v$  range over all terms
- FORT is based on tree automata techniques and (hence) variables range over ground terms
- confluence  $\neq$  ground-confluence

## FSCD 2016 submission

*It should be stressed that the above properties are restricted to ground terms. So CR stands for ground-confluence, which is different from confluence, even in the presence of ground terms; consider e.g. the rules  $f(x) \rightarrow x$  and  $f(x) \rightarrow c$ .*

## Definition

TRS is confluent if  $\forall s \forall t \forall u (s \rightarrow^* t \wedge s \rightarrow^* u \implies \exists v (t \rightarrow^* v \wedge u \rightarrow^* v))$

## Remarks

- variables  $s, t, u, v$  range over all terms
- FORT is based on tree automata techniques and (hence) variables range over ground terms
- confluence  $\neq$  ground-confluence

## FSCD 2016 submission

*It should be stressed that the above properties are restricted to ground terms. So CR stands for ground-confluence, which is different from confluence, even in the presence of ground terms; consider e.g. the rules  $f(x) \rightarrow x$  and  $f(x) \rightarrow c$ . For (left-linear) right-ground TRSs there is no difference.*

## Example

TRS

$$a \rightarrow b$$

$$f(a, x) \rightarrow b$$

$$f(b, b) \rightarrow b$$

## Example

TRS

$$a \rightarrow b$$

$$f(a, x) \rightarrow b$$

$$f(b, b) \rightarrow b$$

is ground-confluent but not confluent

## Example

TRS

$$a \rightarrow b$$

$$f(a, x) \rightarrow b$$

$$f(b, b) \rightarrow b$$

is ground-confluent but not confluent

## Confluence Related Properties

$$\text{CR: } \forall s \forall t \forall u (s \rightarrow^* t \wedge s \rightarrow u \implies t \downarrow u)$$

$$\text{WCR: } \forall s \forall t \forall u (s \rightarrow t \wedge s \rightarrow u \implies t \downarrow u)$$

$$\text{UN: } \forall s \forall t \forall u (s \rightarrow^! t \wedge s \rightarrow^! u \implies t = u)$$

$$\text{UNC: } \forall t \forall u (t \leftrightarrow^* u \wedge \text{NF}(t) \wedge \text{NF}(u) \implies t = u)$$

$$\text{NFP: } \forall s \forall t \forall u (s \rightarrow t \wedge s \rightarrow^! u \implies t \rightarrow^! u)$$

$$\text{SCR: } \forall s \forall t \forall u (s \rightarrow t \wedge s \rightarrow u \implies \exists v (t \rightarrow^= v \wedge u \rightarrow^* v))$$



## Example

TRS

$$a \rightarrow b$$

$$f(a, x) \rightarrow b$$

$$f(b, b) \rightarrow b$$

is ground-confluent but not confluent

## Confluence Related Properties

$$\text{CR: } \forall s \forall t \forall u (s \rightarrow^* t \wedge s \rightarrow u \implies t \downarrow u)$$

$$\text{WCR: } \forall s \forall t \forall u (s \rightarrow t \wedge s \rightarrow u \implies t \downarrow u)$$

$$\text{UN: } \forall s \forall t \forall u (s \rightarrow^! t \wedge s \rightarrow^! u \implies t = u)$$

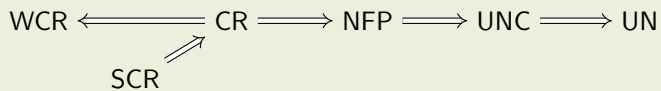
$$\text{UNC: } \forall t \forall u (t \leftrightarrow^* u \wedge \text{NF}(t) \wedge \text{NF}(u) \implies t = u)$$

$$\text{NFP: } \forall s \forall t \forall u (s \rightarrow t \wedge s \rightarrow^! u \implies t \rightarrow^! u)$$

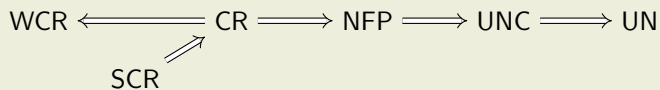
$$\text{SCR: } \forall s \forall t \forall u (s \rightarrow t \wedge s \rightarrow u \implies \exists v (t \rightarrow^= v \wedge u \rightarrow^* v))$$

$$\mathfrak{P} = \{ \text{CR}, \text{WCR}, \text{UN}, \text{UNC}, \text{NFP}, \text{SCR} \}$$

## Relationships



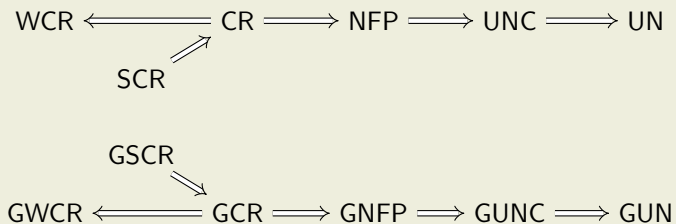
## Relationships



## Notation

$GP$  denotes property  $P$  restricted to ground terms

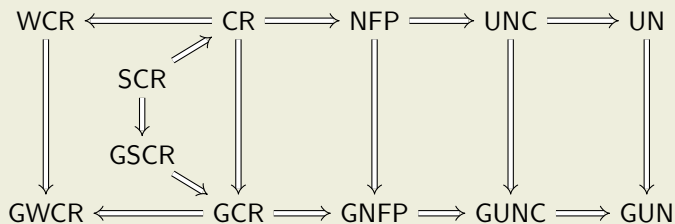
## Relationships



## Notation

$GP$  denotes property  $P$  restricted to ground terms

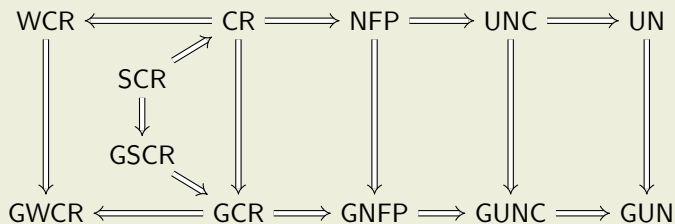
## Relationships



## Notation

$GP$  denotes property  $P$  restricted to ground terms

## Relationships



## Notation

$GP$  denotes property  $P$  restricted to ground terms

## Remark

$\forall P \in \mathfrak{P} \quad GP \not\Rightarrow P$

## Lemma

$\forall$  left-linear right-ground TRS  $(\mathcal{F}, \mathcal{R})$

$$\mathbf{1} \quad (\mathcal{F}, \mathcal{R}) \models P \quad \iff \quad (\mathcal{F} \cup \{c\}, \mathcal{R}) \models GP \quad \forall P \in \mathfrak{P} \setminus \{\text{UNC}\}$$

with fresh constant  $c$

## Lemma

$\forall$  left-linear right-ground TRS  $(\mathcal{F}, \mathcal{R})$

$$\mathbf{1} \quad (\mathcal{F}, \mathcal{R}) \models P \iff (\mathcal{F} \cup \{c\}, \mathcal{R}) \models GP \quad \forall P \in \mathfrak{P} \setminus \{\text{UNC}\}$$

$$\mathbf{2} \quad (\mathcal{F}, \mathcal{R}) \models \text{UNC} \iff (\mathcal{F} \cup \{c, c'\}, \mathcal{R}) \models \text{GUNC}$$

with fresh constants  $c$  and  $c'$



## Lemma

$\forall$  left-linear right-ground TRS  $(\mathcal{F}, \mathcal{R})$

$$\mathbf{1} \quad (\mathcal{F}, \mathcal{R}) \models P \iff (\mathcal{F} \cup \{c\}, \mathcal{R}) \models GP \quad \forall P \in \mathfrak{P} \setminus \{\text{UNC}\}$$

$$\mathbf{2} \quad (\mathcal{F}, \mathcal{R}) \models \text{UNC} \iff (\mathcal{F} \cup \{c, c'\}, \mathcal{R}) \models \text{GUNC}$$

with fresh constants  $c$  and  $c'$

## Example

left-linear right-ground TRS

$$a \rightarrow b \quad f(x, a) \rightarrow f(b, b) \quad f(b, x) \rightarrow f(b, b) \quad f(f(x, y), z) \rightarrow f(b, b)$$

- does not satisfy UNC:  $f(x, b) \leftarrow f(x, a) \rightarrow f(b, b) \leftarrow f(y, a) \rightarrow f(y, b)$

## Lemma

$\forall$  left-linear right-ground TRS  $(\mathcal{F}, \mathcal{R})$

$$1 \quad (\mathcal{F}, \mathcal{R}) \models P \iff (\mathcal{F} \cup \{c\}, \mathcal{R}) \models GP \quad \forall P \in \mathfrak{P} \setminus \{\text{UNC}\}$$

$$2 \quad (\mathcal{F}, \mathcal{R}) \models \text{UNC} \iff (\mathcal{F} \cup \{c, c'\}, \mathcal{R}) \models \text{GUNC}$$

with fresh constants  $c$  and  $c'$

## Example

left-linear right-ground TRS

$$a \rightarrow b \quad f(x, a) \rightarrow f(b, b) \quad f(b, x) \rightarrow f(b, b) \quad f(f(x, y), z) \rightarrow f(b, b)$$

- does not satisfy UNC:  $f(x, b) \leftarrow f(x, a) \rightarrow f(b, b) \leftarrow f(y, a) \rightarrow f(y, b)$
- adding single fresh constant  $c$  is not enough to violate GUNC

## Lemma

$\forall$  left-linear right-ground TRS  $(\mathcal{F}, \mathcal{R})$

$$1 \quad (\mathcal{F}, \mathcal{R}) \models P \iff (\mathcal{F} \cup \{c\}, \mathcal{R}) \models GP \quad \forall P \in \mathfrak{P} \setminus \{\text{UNC}\}$$

$$2 \quad (\mathcal{F}, \mathcal{R}) \models \text{UNC} \iff (\mathcal{F} \cup \{c, c'\}, \mathcal{R}) \models \text{GUNC}$$

with fresh constants  $c$  and  $c'$

## Example

left-linear right-ground TRS

$$a \rightarrow b \quad f(x, a) \rightarrow f(b, b) \quad f(b, x) \rightarrow f(b, b) \quad f(f(x, y), z) \rightarrow f(b, b)$$

- does not satisfy UNC:  $f(x, b) \leftarrow f(x, a) \rightarrow f(b, b) \leftarrow f(y, a) \rightarrow f(y, b)$
- adding single fresh constant  $c$  is not enough to violate GUNC
- GUNC is violated by adding another fresh constant  $c'$ :

$$f(c, b) \leftarrow f(c, a) \rightarrow f(b, b) \leftarrow f(c', a) \rightarrow f(c', b)$$

## Remarks

- for termination (SN) and normalization (WN) no fresh constants are needed

## Remarks

- for termination (SN) and normalization (WN) no fresh constants are needed
- for **other properties** expressible in first-order theory of rewriting adding one or two constants may be insufficient

## Remarks

- for termination (SN) and normalization (WN) no fresh constants are needed
- for **other properties** expressible in first-order theory of rewriting adding one or two constants may be insufficient:

TRS consisting of rule  $f(x) \rightarrow a$  satisfies

$$P: \quad \neg \exists s \exists t \exists u \forall v (v \leftrightarrow^* s \vee v \leftrightarrow^* t \vee v \leftrightarrow^* u)$$

## Remarks

- for termination (SN) and normalization (WN) no fresh constants are needed
- for **other properties** expressible in first-order theory of rewriting adding one or two constants may be insufficient:

TRS consisting of rule  $f(x) \rightarrow a$  satisfies

$$P: \quad \neg \exists s \exists t \exists u \forall v (v \leftrightarrow^* s \vee v \leftrightarrow^* t \vee v \leftrightarrow^* u)$$

but **GP does not hold** after adding additional constants  $c$  and  $c'$

## Remarks

- for termination (SN) and normalization (WN) no fresh constants are needed
- for other properties expressible in first-order theory of rewriting adding one or two constants may be insufficient:

TRS consisting of rule  $f(x) \rightarrow a$  satisfies

$$P: \quad \neg \exists s \exists t \exists u \forall v (v \leftrightarrow^* s \vee v \leftrightarrow^* t \vee v \leftrightarrow^* u)$$

but *GP* does not hold after adding additional constants  $c$  and  $c'$

- adding fresh **unary** function symbol  $g$  and fresh **constant**  $c$  in order to create infinitely many ground normal forms



## Remarks

- for termination (SN) and normalization (WN) no fresh constants are needed
- for other properties expressible in first-order theory of rewriting adding one or two constants may be insufficient:

TRS consisting of rule  $f(x) \rightarrow a$  satisfies

$$P: \quad \neg \exists s \exists t \exists u \forall v (v \leftrightarrow^* s \vee v \leftrightarrow^* t \vee v \leftrightarrow^* u)$$

but *GP* does not hold after adding additional constants  $c$  and  $c'$

- adding fresh **unary** function symbol  $g$  and fresh **constant**  $c$  in order to create infinitely many ground normal forms is **unsound** in general:

consider  $a \rightarrow b$  and  $\forall s \forall t (s \rightarrow t \implies s \xrightarrow{\epsilon} t)$

## Definition

signature  $\mathcal{F}$  is **monadic** if  $\mathcal{F}$  contains no function symbols of arity  $> 1$

## Definition

signature  $\mathcal{F}$  is monadic if  $\mathcal{F}$  contains no function symbols of arity  $> 1$

## Lemma

$\forall$  left-linear right-ground TRS  $(\mathcal{F}, \mathcal{R})$  such that  $\mathcal{R}$  is ground or  $\mathcal{F}$  is monadic

$$(\mathcal{F}, \mathcal{R}) \models P \iff (\mathcal{F}, \mathcal{R}) \models GP \quad \forall P \in \mathfrak{P}$$

## Definition

signature  $\mathcal{F}$  is monadic if  $\mathcal{F}$  contains no function symbols of arity  $> 1$

## Lemma

$\forall$  left-linear right-ground TRS  $(\mathcal{F}, \mathcal{R})$  such that  $\mathcal{R}$  is ground or  $\mathcal{F}$  is monadic

$$(\mathcal{F}, \mathcal{R}) \models P \iff (\mathcal{F}, \mathcal{R}) \models GP \quad \forall P \in \mathfrak{P}$$

## Example

checking GCR of TRS

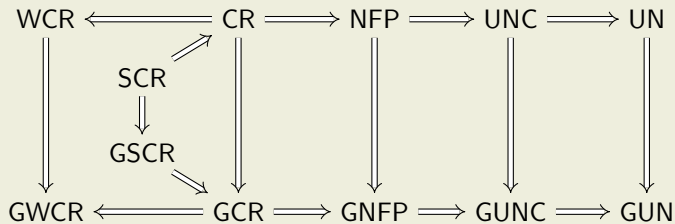
$$\begin{array}{lll} f(f(f(x))) \rightarrow a & f(f(a)) \rightarrow a & f(a) \rightarrow a \\ f(f(g(g(x)))) \rightarrow f(a) & g(f(a)) \rightarrow a & g(a) \rightarrow a \end{array}$$

takes 0.85 seconds but 1.73 seconds if fresh constant is added

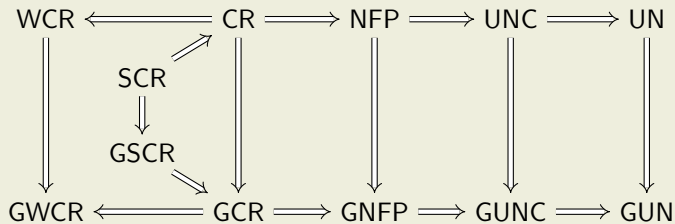
# Outline

- First-Order Theory of Rewriting
- Automation
- Properties on Open Terms
- **Experiments**
- Future Work

## Relationships



## Relationships

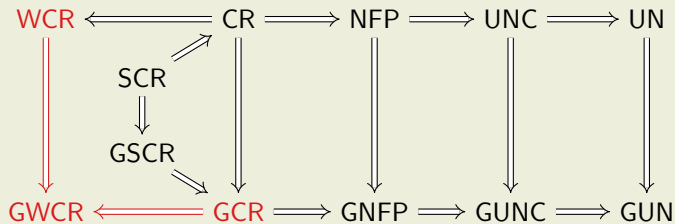


## Synthesis Experiments with FORT 0.2



```
-s -f "a:0 b:0 f:2"
```

## Relationships

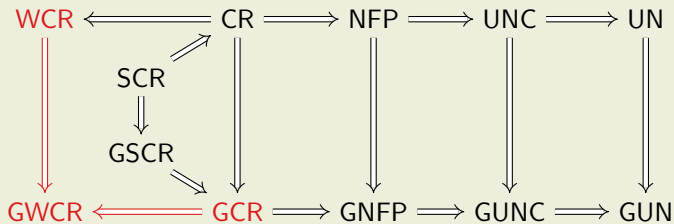
Synthesis Experiments with **FORT 0.2**

```
-S -f "a:0 b:0 f:2"
```


```
"GWCR & ~WCR & ~GCR"
```



## Relationships

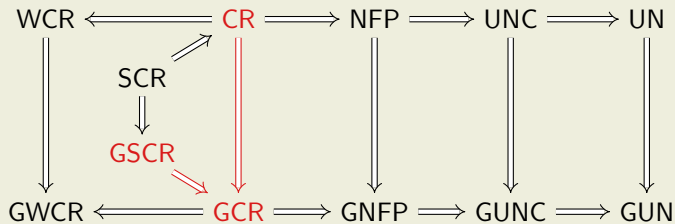


## Synthesis Experiments with FORT 0.2


 -S -f "a:0 b:0 f:2"

"GWCR & ~WCR & ~GCR"     $a \rightarrow b$      $f(x, a) \rightarrow a$      $a \rightarrow f(a, a)$     80 s

## Relationships



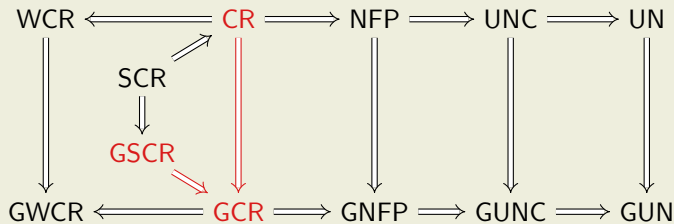
## Synthesis Experiments with FORT 0.2

 -S -f "a:0 b:0 f:2"


"GWCR & ~WCR & ~GCR"     $a \rightarrow b$      $f(x, a) \rightarrow a$      $a \rightarrow f(a, a)$     80 s

"GCR & ~CR & ~GSCR"

## Relationships



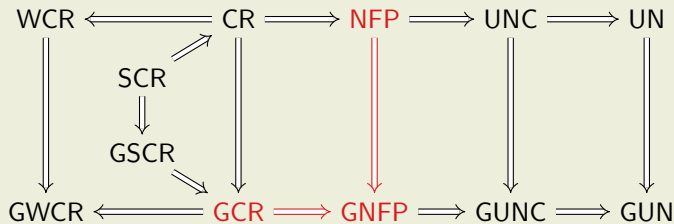
## Synthesis Experiments with FORT 0.2

 -S -f "a:0 b:0 f:2"


"GWCR & ~WCR & ~GCR"     $a \rightarrow b$      $f(x, a) \rightarrow a$      $a \rightarrow f(a, a)$     80 s

"GCR & ~CR & ~GSCR"     $a \rightarrow b$      $f(x, a) \rightarrow b$      $b \rightarrow f(a, a)$     109 s

## Relationships



## Synthesis Experiments with FORT 0.2

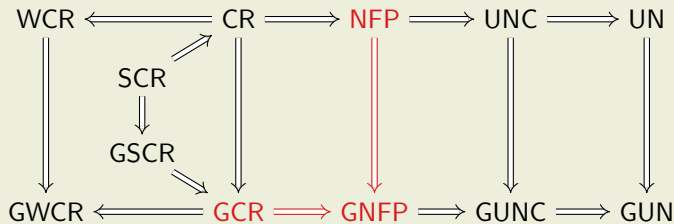
 -S -f "a:0 b:0 f:2"

"GWCR & ~WCR & ~GCR"     $a \rightarrow b$      $f(x, a) \rightarrow a$      $a \rightarrow f(a, a)$     80 s

"GCR & ~CR & ~GSCR"     $a \rightarrow b$      $f(x, a) \rightarrow b$      $b \rightarrow f(a, a)$     109 s

"GNFP & ~NFP & ~GCR"

## Relationships



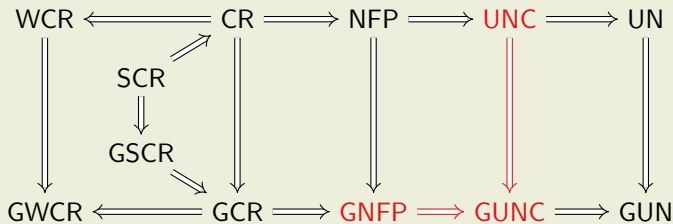
## Synthesis Experiments with FORT 0.2




`-S -f "a:0 b:0 f:2"`

"GWCR & ~WCR & ~GCR"	$a \rightarrow b$	$f(x, a) \rightarrow a$	$a \rightarrow f(a, a)$	80 s
"GCR & ~CR & ~GSCR"	$a \rightarrow b$	$f(x, a) \rightarrow b$	$b \rightarrow f(a, a)$	109 s
"GNFP & ~NFP & ~GCR"	$a \rightarrow b$	$f(x, a) \rightarrow f(a, a)$	$f(b, b) \rightarrow f(a, a)$	16 s

## Relationships



## Synthesis Experiments with FORT 0.2

 -S -f "a:0 b:0 f:2"

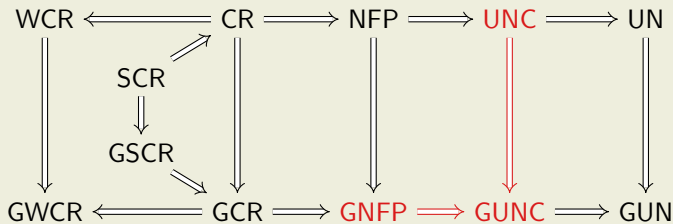
"GWCR & ~WCR & ~GCR"     $a \rightarrow b$      $f(x, a) \rightarrow a$      $a \rightarrow f(a, a)$     80 s

"GCR & ~CR & ~GSCR"     $a \rightarrow b$      $f(x, a) \rightarrow b$      $b \rightarrow f(a, a)$     109 s


"GNFP & ~NFP & ~GCR"     $a \rightarrow b$      $f(x, a) \rightarrow f(a, a)$      $f(b, b) \rightarrow f(a, a)$     16 s

"GUNC & ~UNC & ~GNFP"

## Relationships

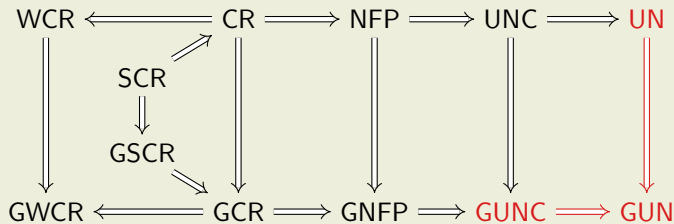


## Synthesis Experiments with FORT 0.2


 -S -f "a:0 b:0 f:2"

"GWCR & ~WCR & ~GCR"	$a \rightarrow b$	$f(x, a) \rightarrow a$	$a \rightarrow f(a, a)$	80 s
"GCR & ~CR & ~GSCR"	$a \rightarrow b$	$f(x, a) \rightarrow b$	$b \rightarrow f(a, a)$	109 s
"GNFP & ~NFP & ~GCR"	$a \rightarrow b$	$f(x, a) \rightarrow f(a, a)$	$f(b, b) \rightarrow f(a, a)$	16 s
"GUNC & ~UNC & ~GNFP"	$a \rightarrow a$	$f(x, a) \rightarrow a$	$f(b, x) \rightarrow b$	95 s

## Relationships



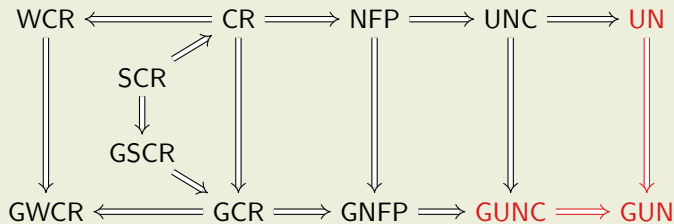
## Synthesis Experiments with FORT 0.2

 -S -f "a:0 b:0 f:2"


"GWCR & ~WCR & ~GCR"	$a \rightarrow b$	$f(x, a) \rightarrow a$	$a \rightarrow f(a, a)$	80 s
"GCR & ~CR & ~GSCR"	$a \rightarrow b$	$f(x, a) \rightarrow b$	$b \rightarrow f(a, a)$	109 s
"GNFP & ~NFP & ~GCR"	$a \rightarrow b$	$f(x, a) \rightarrow f(a, a)$	$f(b, b) \rightarrow f(a, a)$	16 s
"GUNC & ~UNC & ~GNFP"	$a \rightarrow a$	$f(x, a) \rightarrow a$	$f(b, x) \rightarrow b$	95 s



## Relationships

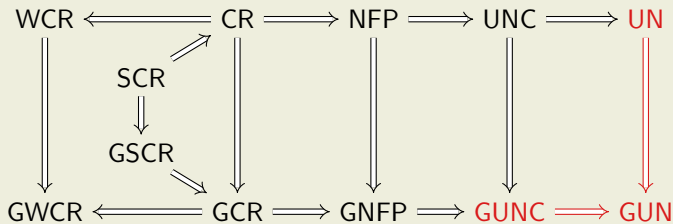



## Synthesis Experiments with FORT 0.2

 -S -f "a:0 b:0 f:2"

"GWCR & ~WCR & ~GCR"	$a \rightarrow b$	$f(x, a) \rightarrow a$	$a \rightarrow f(a, a)$	80 s
"GCR & ~CR & ~GSCR"	$a \rightarrow b$	$f(x, a) \rightarrow b$	$b \rightarrow f(a, a)$	109 s
"GNFP & ~NFP & ~GCR"	$a \rightarrow b$	$f(x, a) \rightarrow f(a, a)$	$f(b, b) \rightarrow f(a, a)$	16 s
"~GNFP & GUNC & ~UNC"	$a \rightarrow a$	$f(x, a) \rightarrow a$	$f(b, x) \rightarrow b$	21 s

## Relationships

Synthesis Experiments with **FORT 1.0**

 -S -f "a:0 b:0 f:2"

"GWCR & ~WCR & ~GCR"	$a \rightarrow b$	$f(x, a) \rightarrow a$	$a \rightarrow f(a, a)$	8-20 s
"GCR & ~CR & ~GSCR"	$a \rightarrow b$	$f(x, a) \rightarrow b$	$b \rightarrow f(a, a)$	8-10 s
"GNFP & ~NFP & ~GCR"	$a \rightarrow b$	$f(x, a) \rightarrow f(a, a)$	$f(b, b) \rightarrow f(a, a)$	9-11 s
"~GNFP & GUNC & ~UNC"	$a \rightarrow a$	$f(x, a) \rightarrow a$	$f(b, x) \rightarrow b$	2-4 s

## Example

"GUN & ~UN & ~GUNC"

$$b \rightarrow a$$

$$b \rightarrow c$$

$$c \rightarrow c$$

$$d \rightarrow c$$

$$d \rightarrow e$$

$$f(x, a) \rightarrow A$$

$$f(x, e) \rightarrow A$$

$$f(x, A) \rightarrow A$$

$$f(c, x) \rightarrow A$$

## Example

"GUN & ~UN & ~GUNC"

$$b \rightarrow a$$

$$d \rightarrow c$$

$$f(x, e) \rightarrow A$$

$$b \rightarrow c$$

$$d \rightarrow e$$

$$f(x, A) \rightarrow A$$

$$c \rightarrow c$$

$$f(x, a) \rightarrow A$$

$$f(c, x) \rightarrow A$$

## Comparison (GCR)

**AGCP** is recent tool for **ground-confluence** of many-sorted TRSs based on rewriting induction (Aoto and Toyama, FSCD 2016)

## Example

"GUN &amp; ~UN &amp; ~GUNC"

$b \rightarrow a$

$d \rightarrow c$

$f(x, e) \rightarrow A$

$b \rightarrow c$

$d \rightarrow e$

$f(x, A) \rightarrow A$

$c \rightarrow c$

$f(x, a) \rightarrow A$

$f(c, x) \rightarrow A$

## Comparison (GCR)

AGCP is recent tool for ground-confluence of many-sorted TRSs based on rewriting induction (Aoto and Toyama, FSCD 2016)

65 TRSs	AGCP ( $\emptyset$ time)	FORT 0.2 ( $\emptyset$ time)
yes	8 (0.02 s)	42 (0.42 s)
no	–	14 (3.88 s)
maybe	56 (0.19 s)	–
timeout	1	9
total time	71 s	612 s

## Example

"GUN &amp; ~UN &amp; ~GUNC"

$b \rightarrow a$

$d \rightarrow c$

$f(x, e) \rightarrow A$

$b \rightarrow c$

$d \rightarrow e$

$f(x, A) \rightarrow A$

$c \rightarrow c$

$f(x, a) \rightarrow A$

$f(c, x) \rightarrow A$

## Comparison (GCR)

AGCP is recent tool for ground-confluence of many-sorted TRSs based on rewriting induction (Aoto and Toyama, FSCD 2016)

65 TRSs	AGCP ( $\emptyset$ time)	FORT 0.2 ( $\emptyset$ time)	FORT 1.0 ( $\emptyset$ time)
yes	8 (0.02 s)	42 (0.42 s)	43 (0.26 s)
no	–	14 (3.88 s)	18 (0.96 s)
maybe	56 (0.19 s)	–	–
timeout	1	9	4
total time	71 s	612 s	268 s

## Example

"GUN &amp; ~UN &amp; ~G"

Let's see what happens at



$$f(x, e) \rightarrow A$$

$$f(x, A) \rightarrow A$$

$$f(c, x) \rightarrow A$$

## Comparison (GCR)

AGCP is recent tool for ground-confluence of **many-sorted** TRSs based on rewriting induction (Aoto and Toyama, FSCD 2016)

65 TRSs	AGCP ( $\emptyset$ time)	FORT 0.2 ( $\emptyset$ time)	FORT 1.0 ( $\emptyset$ time)
yes	8 (0.02 s)	42 (0.42 s)	43 (0.26 s)
no	–	14 (3.88 s)	18 (0.96 s)
maybe	56 (0.19 s)	–	–
timeout	1	9	4
total time	71 s	612 s	268 s

# Outline

- First-Order Theory of Rewriting
- Automation
- Properties on Open Terms
- Experiments
- Future Work



## Limitation

restriction to left-linear right-ground TRSs is hard to overcome because first-order theory of **one-step** rewriting ( $\rightarrow$ ) is undecidable

## Limitation

restriction to left-linear right-ground TRSs is hard to overcome because first-order theory of **one-step** rewriting ( $\rightarrow$ ) is undecidable

- ... even for linear non-erasing TRSs (Treinen, 1998)

## Limitation

restriction to left-linear right-ground TRSs is hard to overcome because first-order theory of **one-step** rewriting ( $\rightarrow$ ) is undecidable

- ... even for linear non-erasing TRSs (Treinen, 1998)
- ... even for complete right-ground TRSs (Vorobyov, 2002)

## Limitation

restriction to left-linear right-ground TRSs is hard to overcome because first-order theory of one-step rewriting ( $\rightarrow$ ) is undecidable

- ... even for linear non-erasing TRSs (Treinen, 1998)
- ... even for complete right-ground TRSs (Vorobyov, 2002)

## Ongoing and Future Work

- **generating witnesses** for existential formulas and formulas with free variables

## Limitation

restriction to left-linear right-ground TRSs is hard to overcome because first-order theory of one-step rewriting ( $\rightarrow$ ) is undecidable

- ... even for linear non-erasing TRSs (Treinen, 1998)
- ... even for complete right-ground TRSs (Vorobyov, 2002)

## Ongoing and Future Work

- generating witnesses for existential formulas and formulas with free variables
- support for **combinations of TRSs** (e.g., to express commutation)

## Limitation

restriction to left-linear right-ground TRSs is hard to overcome because first-order theory of one-step rewriting ( $\rightarrow$ ) is undecidable

- ... even for linear non-erasing TRSs (Treinen, 1998)
- ... even for complete right-ground TRSs (Vorobyov, 2002)

## Ongoing and Future Work

- generating witnesses for existential formulas and formulas with free variables
- support for combinations of TRSs (e.g., to express commutation)
- incorporating rewrite **strategies**

## Limitation

restriction to left-linear right-ground TRSs is hard to overcome because first-order theory of one-step rewriting ( $\rightarrow$ ) is undecidable

- ... even for linear non-erasing TRSs (Treinen, 1998)
- ... even for complete right-ground TRSs (Vorobyov, 2002)

## Ongoing and Future Work

- generating witnesses for existential formulas and formulas with free variables
- support for combinations of TRSs (e.g., to express commutation)
- incorporating rewrite strategies
- **formalizing** underlying theory in Isabelle/HOL