Decreasing proof orders
Interpreting conversions in involutive monoids

Vincent van Oostrom
Universiteit Utrecht

IWC, Nagoya, May 29, 2012
Decreasing tiles

Involutive proofs

French strings

Applications
Alhambra
Tiling puzzles (1964–73)
Tiling puzzles (1964–73)
Tiling puzzles (1964–73)
Tiling puzzles (1964–73)
Tiling puzzles (1964–73)
Tiling puzzles (1964–73)
Tiling puzzles (1964–73)
Tiling puzzles (1964–73)
Tiling puzzles (1964–73)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Scalable tile puzzling (1964–78)
Puzzling tiles... (1942–60)
Puzzling tiles. . . (1942–60)
Puzzling tiles. . . (1942–60)
Puzzling tiles. . . (1942–60)
Puzzling tiles... (1942–60)

Decreasing tiles
Involutive proofs
French strings
Applications
Puzzling tiles... (1942–60)
Puzzling tiles... (1942–60)
Puzzling tiles... (1942–60)
Puzzling tiles... (1942–60)
Puzzling tiles... (1942–60)
Decreasing tiles
Involutive proofs
French strings
Applications

Puzzling tiles... (1942–60)
Puzzling tiles... (1942–60)
Puzzling tiling questions
Puzzling tiling questions

Given a set of tiles:
Puzzling tiling questions

Given a set of tiles:

- For any situation, is there at least one fitting tile?
Puzzling tiling questions

Given a set of tiles:

- For any situation, is there at least one fitting tile?
- Does a tiling strategy exist that terminates?
Puzzling tiling questions

Given a set of tiles:

- For any situation, is there at least one fitting tile?
- Does a tiling strategy exist that terminates?
- Do all tiling strategies terminate?
- How many tiles are needed?
Decreasing tiles (1978–94)
Decreasing tiles (1978–94)
Decreasing tiles (1978–94)
Decreasing tiles (1978–94)
Decreasing tiles (1978–94)
Decreasing tiles (1978–94)
Decreasing tiles (1978–94)
Decreasing tiles (1978–94)

Definition
set of such tiles decreasing if used colours well-ordered
Decreasing tiles (1978–94)

Definition

set of such tiles decreasing if used colours well-founded
Terminating tiling strategy for decreasing tiles

Memorandum 78-08.
Issued August 1978.

A note on weak diamond properties.

by

N.G. de Bruijn.
Terminating tiling strategy for decreasing tiles

Memorandum 78-08.
Issued August 1978.

A note on weak diamond properties.

by

N.G. de Bruijn.

Theorem

if tiles are decreasing, a tiling strategy exists that terminates.
Decreasing rewrite systems
Decreasing rewrite systems
Theorem

*if rewrite system decreasing, then confluent*
Tiles with bite (2008–)
Tiles with bite (2008–)
Tiles with bite (2008–)
Tiles with bite (2008–)
Tiles with bite (2008–)
Tiles with bite (2008–)
Tiles with bite (2008–)
Tiles with bite (2008–)
Tiles with bite (2008–)
Tiles with bite (2008–)
Tiles with bite (2008–)

Decreasing tiles
Involution proofs
French strings
Applications
Decreasing converted rewrite systems

Theorem

If tiles decreasing converted, a tiling strategy exists that terminates
Decreasing converted rewrite systems

Theorem

if rewrite system decreasing converted, then confluent
Given set of decreasing tiles:

- Previous work: terminating tiling strategy exist
Given set of decreasing tiles:
  • Previous work: terminating tiling strategy exist
  • This talk: all tiling strategies terminate
Transforming conversions
Transforming conversions

a convertible to e
Transforming conversions

\[ a \text{ convertible to } e \]
Transforming conversions

a convertible to e
Transforming conversions

a convertible to e
Transforming conversions

a convertible to e
Transforming conversions

a convertible to e
Transforming conversions

a convertible to e by rewrite proof
why do these transformations terminate?
Equational logic on nullary symbols (constants)

\[
\begin{align*}
  a \to b & \quad \text{\(\text{step}\)} \quad a = b \\
  a = b & \quad \text{(e)} \quad a = a \\
  b = a & \quad \text{(-1)} \\
  a = c & \quad \text{(.)} \\
  a = b \quad b = c & \\
  a = b \quad b = c & \\
\end{align*}
\]
Equational logic on nullary symbols (constants)

\[
\begin{align*}
  a & \rightarrow b \\
  \frac{a = b}{(step)} & \quad \frac{a = b}{(e)} & \quad \frac{a = b}{(-1)} & \quad \frac{a = b \quad b = c}{(\cdot)} \\
  a = a & \quad b = a & \quad a = c
\end{align*}
\]

no derivation rules for congruence or substitution
Equational logic on nullary symbols (constants)

\[
a \rightarrow b \\
a = b \quad \text{(step)} \\
a = a \\
b = a \\
a = b \quad b = c \\
a = c \\
\]

no derivation rules for congruence or substitution

Theorem ((sub)Birkhoff)

*abstract rewriting is logical, that is, = coincides with \( \leftrightarrow^* \)*
Equational logic on nullary symbols (constants)

\[
\begin{align*}
  a &\rightarrow b \\
  \frac{a = b}{(\text{step})} & \quad \frac{a = b}{(\text{e})} & \quad \frac{a = b}{(-1)} & \quad \frac{a = b \quad b = c}{(\cdot)}
\end{align*}
\]

no derivation rules for congruence or substitution

Theorem ((sub)Birkhoff)

abstract rewriting is logical, that is, \( = \) coincides with \( \leftrightarrow^* \)

Methodology to show transformation of conversions terminates:
Equational logic on nullary symbols (constants)

\[
\frac{a \rightarrow b}{a = b} \quad (\text{step}) \quad \frac{a = b}{a = a} \quad (e) \quad \frac{a = b}{b = a} \quad (-1) \quad \frac{a = b \quad b = c}{a = c} \quad (\cdot)
\]

no derivation rules for congruence or substitution

Theorem ((sub)Birkhoff)

abstract rewriting is logical, that is, $\equiv$ coincides with $\leftrightarrow^*$

Methodology to show transformation of conversions terminates:

- conversion is proof (in equational logic)
Equational logic on nullary symbols (constants)

\[
\begin{align*}
  a & \rightarrow b \\
  \frac{a = b}{\text{(step)}} & \quad \frac{a = b}{\text{(e)}} & \quad \frac{a = b}{\text{(-1)}} & \quad \frac{a = b \quad b = c}{\cdot} \\
  \frac{a = a}{\text{coincides with}} & \quad \frac{b = a}{\leftrightarrow^*} & \quad \frac{a = c}{\text{Methodology to show transformation of conversions terminates:}}
\end{align*}
\]

- no derivation rules for congruence or substitution

Theorem ((sub)Birkhoff)

abstract rewriting is logical, that is, $\equiv$ coincides with $\leftrightarrow^*$

Methodology to show transformation of conversions terminates:

- conversion is proof (in equational logic)
- represent proof as proof term (term over \{\text{step, } -1, \cdot, e\})
Equational logic on nullary symbols (constants)

\[ a \rightarrow b \quad \frac{a = b}{a = a} \quad \frac{a = b}{b = a} \quad \frac{a = b \quad b = c}{a = c} \]

no derivation rules for congruence or substitution

Theorem ((sub)Birkhoff)

abstract rewriting is logical, that is, = coincides with \( \leftrightarrow^* \)

Methodology to show transformation of conversions terminates:

- conversion is proof (in equational logic)
- represent proof as proof term (term over \{\text{step}, -1, \cdot, e\})
- example: proof term \( m^{-1} \cdot (\ell \cdot (k^{-1} \cdot m)) \) represents conversion \( a \leftarrow_m b \rightarrow_\ell c \leftarrow_k a \rightarrow_m b \)
Equational logic on nullary symbols (constants)

\[
\begin{align*}
  a \rightarrow b & \quad (\text{step}) &
  a\! =\! b & \quad (\text{e}) &
  a\! =\! b \quad (\cdot) & \quad a\! =\! c
\end{align*}
\]

\[
\begin{align*}
  \frac{a\! =\! b}{a\! =\! a} & \quad (\text{e}) &
  \frac{a\! =\! b \quad b\! =\! c}{a\! =\! c}
\end{align*}
\]

no derivation rules for congruence or substitution

Theorem ((sub)Birkhoff)

abstract rewriting is logical, that is, = coincides with $\leftrightarrow^*$

Methodology to show transformation of conversions terminates:

- conversion is proof (in equational logic)
- represent proof as proof term (term over $\{\text{step}, -1, \cdot, e\}$)
- example: proof term $m^{-1} \cdot (\ell \cdot (k^{-1} \cdot m))$ represents conversion $a \leftarrow_m b \rightarrow_{\ell} c \leftarrow_k a \rightarrow_m b$
- equip proof terms with terminating rewrite relation compatible with decreasingness
Conversions → proof terms → involutive monoid

Definition

set with

- associative binary operation \( \cdot \)
- identity element \( e \)
- involutive anti-automorphism \( -1 \)

\[
\begin{align*}
(a \cdot b) \cdot c & = a \cdot (b \cdot c) \quad \text{(associative)} \\
 a \cdot e & = a \quad \text{(right identity)} \\
 e \cdot a & = a \quad \text{(left identity)} \\
 (a^{-1})^{-1} & = a \quad \text{(involutary)} \\
 (a \cdot b)^{-1} & = b^{-1} \cdot a^{-1} \quad \text{(anti-automorphic)} \\
 \varepsilon^{-1} & = \varepsilon \quad \text{(derived)}
\end{align*}
\]
Involutive monoid examples

- \{\ast\} with binary, nullary, unary constant-\ast map
Involution monoid examples

- $\{\ast\}$ with binary, nullary, unary constant-$\ast$ map
- integers with addition, zero, unary minus
Involution monoid examples

- $\{\ast\}$ with binary, nullary, unary constant-$\ast$ map
- positive rationals with multiplication, one, inverse
Involution monoid examples

- \(\{\ast\}\) with binary, nullary, unary constant-\(\ast\) map
- group
Involution monoid examples

- $\{\star\}$ with binary, nullary, unary constant-$\star$ map
- group (examples $(\mathbb{Z}, +, 0, -)$, $(\mathbb{Q}^+, \cdot, 1, -1)$)
- natural numbers with addition, zero, identity map
Involution monoid examples

- \{\star\} with binary, nullary, unary constant-\star map
- group (examples (\mathbb{Z}, +, 0, -), (\mathbb{Q}^+, \cdot, 1, -1))
- multisets with multiset sum, empty multiset, identity map
Involution monoid examples

- $\{\ast\}$ with binary, nullary, unary constant-$\ast$ map
- group (examples $(\mathbb{Z}, +, 0, -)$, $(\mathbb{Q}^+, \cdot, 1, -1)$)
- commutative monoid with identity map
Involution monoid examples

- \{\ast\} with binary, nullary, unary constant-\ast map
- group (examples \((\mathbb{Z}, +, 0, -), (\mathbb{Q}^+, \cdot, 1, -1))\)
- commutative monoid (examples \((\mathbb{N}, +, 0), ([L], \oplus, [ ])\))
- diagrams of \ with gluing, point, mirroring in vertical axis
Involution monoid examples

- \{ \ast \} with binary, nullary, unary constant-\ast map
- group (examples \((\mathbb{Z}, +, 0, -)\), \((\mathbb{Q}^+, \cdot, 1, -1)\))
- commutative monoid (examples \((\mathbb{N}, +, 0)\), \(([L], \uplus, [ ]))\)
- diagrams of \(\setminus\) with gluing, point, mirroring in vertical axis
- number pairs with pointwise addition, \((0, 0)\), swapping
Involutive monoid examples

- \{\ast\} with binary, nullary, unary constant-\ast map
- group (examples \((\mathbb{Z}, +, 0, -)\), \((\mathbb{Q}^+, \cdot, 1, -1)\))
- commutative monoid (examples \((\mathbb{N}, +, 0)\), \([L], \lor, [ ]\))
- diagrams of \ with gluing, point, mirroring in vertical axis
- number triples with composition given by
  \((n_1, m_1, k_1) \cdot (n_2, m_2, k_2) = (n_1 + n_2, m_1 + k_1 \cdot n_2 + m_2, k_1 + k_2)\),
  zero \((0, 0, 0)\), involution \((n, m, k)^{-1} = (k, m, n)\)
Involutive monoid examples

- \( \{\ast\} \) with binary, nullary, unary constant-\( \ast \) map
- group (examples \((\mathbb{Z}, +, 0, -), (\mathbb{Q}^+, \cdot, 1, -1)\))
- commutative monoid (examples \((\mathbb{N}, +, 0), ([L], \uplus, [ ]))\))
- diagrams of \( \backslash \) with gluing, point, mirroring in vertical axis
- number triples with composition given by
  \[ (n_1, m_1, k_1) \cdot (n_2, m_2, k_2) = (n_1 + n_2, m_1 + k_1 \cdot n_2 + m_2, k_1 + k_2), \]
  zero \((0, 0, 0)\), involution \((n, m, k)^{-1} = (k, m, n)\)

\[
((n_1, m_1, k_1) \cdot (n_2, m_2, k_2)) \cdot (n_3, m_3, k_3)
= (n_1 + n_2, m_1 + k_1 \cdot n_2 + m_2, k_1 + k_2) \cdot (n_3, m_3, k_3)
= (n_1 + n_2 + n_3, m_1 + k_1 \cdot n_2 + m_2 + (k_1 + k_2) \cdot n_3 + m_3, k_1 + k_2 + k_3)
= (n_1 + n_2 + n_3, m_1 + k_1 \cdot (n_2 + n_3) + m_2 + k_2 \cdot n_3 + m_3, k_1 + k_2 + k_3)
= (n_1, m_1, k_1) \cdot (n_2 + n_3, m_2 + k_2 \cdot n_3 + m_3, k_2 + k_3)
= (n_1, m_1, k_1) \cdot ((n_2, m_2, k_2) \cdot (n_3, m_3, k_3))
\]
Involution monoid of French strings

Definition

- French letter is an accented (acute or grave) letter
Involution monoid of French strings

**Definition**

- **French** letter is an accented (acute or grave) letter
- **juxtaposition** `ëvèn` juxtaposed to `knikté` gives `ëvènkñikté`
Involutive monoid of French strings

Definition

- French letter is an accented (acute or grave) letter
- juxtaposition ·
- empty string $\varepsilon$
Involution monoid of French strings

**Definition**

- **French** letter is an accented (acute or grave) letter
- juxtaposition \( \_ \)
- empty string \( \varepsilon \)
- mirroring \( t \text{èl}k\text{è}n\text{s} \text{m}i\text{r}o\text{r}s \text{s}n\text{è}k\text{l}l\text{è}t \)
Involution monoid of French strings

Definition

- **French** letter is an accented (acute or grave) letter
- juxtaposition \( \_ \)
- empty string \( \varepsilon \)
- mirroring \( \overline{-1} \)
- \( \hat{L} \) set of French Strings on \( L \) (à for either à or á)
Involution monoid of French strings

Definition

- **French** letter is an accented (acute or grave) letter
- juxtaposition
- empty string $\varepsilon$
- mirroring $-1$
- $\hat{L}$ set of French Strings on $L$
Involutive monoid of French strings

Definition

- **French** letter is an accented (acute or grave) letter
- juxtaposition \(_{\cdot}\)
- empty string \(\varepsilon\)
- mirroring \(-1\)
- \(\hat{L}\) set of French Strings on \(L\)

letter markup (representation preserves length, prefix, suffix)
Boustrophedon

Gortyn code, Crete, 5th century B.C. (wikipedia)
Boustrophedon
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
how the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

Involutive proofs

Decreasing tiles

French strings

Applications

How the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the plow coughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

How the cow ploughs
Boustrophedon

how the cow ploughs
Boustrophedon

Martinus Nijhoff, Het kind en ik, Nieuwe Gedichten, 1934 (Hortus Botanicus, Universiteitsmuseum Utrecht, next to pond)
EN TELKENS ALS IK EVEN KNIKT DAT IK HET WIST LIET HIJ HET WATER BEVEN EN HET WERD WITGEWIST
Involution monoid homomorphisms

Definition

Homomorphism is map preserving operations.

Examples

- involutive monoid to itself (identity)
Involutie monoid homomorphisms

Definition

homomorphism is a map preserving operations

Examples

- involutive monoid to itself (identity)
- French strings → number pairs (grave, acute)
  \[ \text{êêñâà} \mapsto (3, 2) \]
Involution monoid homomorphisms

Definition
homomorphism is a map preserving operations

Examples
- involutive monoid to itself (identity)
- number pairs $\rightarrow$ natural numbers (sum)
  $(3, 2) \mapsto 5$
Involution monoid homomorphisms

Definition

homomorphism is map preserving operations

Examples

- involutive monoid to itself (identity)
- French strings $\rightarrow$ natural numbers (length)
  composition of previous two
Involution monoid homomorphisms

Definition
homomorphism is map preserving operations

Examples
- involutive monoid to itself (identity)
- French strings → natural numbers (length)
- French strings → multisets (letters)
  \[ \text{bàřbàřó} \mapsto [a, a, b, b, o, r, r] \]
Involution monoid homomorphisms

**Definition**

Homomorphism is a map preserving operations.

**Examples**

- involutive monoid to itself (identity)
- French strings $\rightarrow$ natural numbers (length)
- French strings $\rightarrow$ multisets (letters)
- French strings $\rightarrow$ diagrams

ćěńàř $\mapsto$
Involution monoid homomorphisms

Definition

A homomorphism is a map preserving operations.

Examples

- involutive monoid to itself (identity)
- French strings \(\rightarrow\) natural numbers (length)
- French strings \(\rightarrow\) multisets (letters)
- diagrams \(\rightarrow\) triples
  \[
  \begin{array}{ccc}
  3 & 5 & 2 \\
  \end{array}
  \]
  \[\mapsto (3, 5, 2)\text{ cf.}
  \]
Involuting monoid homomorphisms

Definition

Homomorphism is a map preserving operations.

Examples

- involutive monoid to itself (identity)
- French strings \(\rightarrow\) natural numbers (length)
- French strings \(\rightarrow\) multisets (letters)
- French strings \(\rightarrow\) triples (area)
  composition of previous two
Free involutive monoid on generators

Theorem

*French strings on* $L$ *give* free involutive monoid on $L$
Free involutive monoid on generators

Theorem

*French strings on* \( L \) *give free involutive monoid on* \( L \)

French string : conversion = string : reduction
Freeness of involutive monoid of French Strings
Freeness of involutive monoid of French Strings

\[ f: L \rightarrow M \]

\[ \text{forget} : M \rightarrow M, c, e, i \]
Freeness of involutive monoid of French Strings

\[ L \rightarrow f \rightarrow M \rightarrow \hat{L}, \varepsilon, -1 \rightarrow M, c, e, i \]

- **SET**
- **INVOLUTIVE MONOID**

- **enrich**
- **forget**
Freeness of involutive monoid of French Strings

\[ \hat{L}, \epsilon, \varepsilon, -1 \]

\[ M, c, e, i \]

\[ \text{SET} \quad \text{ENRICH} \quad \text{INVOLUTIVE MONOID} \]

\[ L \quad \hat{L} \quad \hat{L}, \epsilon, \varepsilon, -1 \]

\[ f \quad \text{forget} \]

\[ M \]

Decreasing tiles
Involutive proofs
French strings
Applications
Freeness of involutive monoid of French Strings

\[ \overset{f}{\longrightarrow} \]

\[ \overset{\ell \leftrightarrow \ell}{\longrightarrow} \]

\[ \overset{\text{enrich}}{\longrightarrow} \]

\[ \overset{\text{forget}}{\longrightarrow} \]

\[ \overset{\hat{L}, \omega, \varepsilon, \ -1}{\longrightarrow} \]

\[ \overset{M, c, e, i}{\longrightarrow} \]

\[ L \]

\[ \hat{L} \]

\[ M \]
Freeness of involutive monoid of French Strings

![Diagram showing the relationship between sets and involutive monoids.]

- SET
- INVOLUTIVE MONOID

\[ L \xrightarrow{\ell \mapsto \hat{\ell}} \hat{L} \xrightarrow{f} M \xrightarrow{\text{forget}} M, c, e, i \]

\[ \hat{L}, \omega, \varepsilon, -1 \]

- \( \exists! \hat{f} \)
- \( \hat{f} \)
- \( f \)
- \( \ell \mapsto \hat{\ell} \)

Decreasing tiles
Involutive proofs
French strings
Applications
Free involutive monoid on generators

Theorem

*French strings on $L$ give free involutive monoid on $L$*
Free involutive monoid on generators

Theorem

French strings on $L$ give free involutive monoid on $L$

Proof.

$\widehat{L}$ in bijection via $\ell \mapsto \ell$, with union of $\{e\}$ and

$$N ::= \ell \mid i(\ell) \mid c(\ell, N) \mid c(i(\ell), N)$$
Free involutive monoid on generators

Theorem

French strings on $L$ give free involutive monoid on $L$

Proof.

$\hat{L}$ in bijection via $\ell \mapsto \ell$, with union of $\{e\}$ and

$$N ::= \ell \mid i(\ell) \mid c(\ell, N) \mid c(i(\ell), N)$$

$N$ set of normal forms on $L$ for TRS completing axioms

- $c(c(x, y), z) \rightarrow c(x, c(y, z))$
- $c(x, e) \rightarrow x$
- $c(e, x) \rightarrow x$
- $i(i(x)) \rightarrow x$
- $i(c(x, y)) \rightarrow c(i(y), i(x))$
- $i(e) \rightarrow e$
Involutive monoid on French terms $L^\sharp$

Definition

**certain terms on certain** French strings
Involutive monoid on French terms $L^\#$

Definition

terms on strings

\[ \begin{array}{c}
\varepsilon & \ell & \varepsilon \\
\varepsilon & \varepsilon & \varepsilon \\
\varepsilon & \varepsilon & \varepsilon \\
\end{array} \]

\[ \text{inorder} \]

\[ mk\ell m \]
Involutive monoid on French terms $L^\#$

Definition

terms on strings

\[ m \quad m \]
\[ \varepsilon \quad k \quad \ell \quad \varepsilon \]
\[ \varepsilon \quad \varepsilon \quad \varepsilon \quad \varepsilon \]

inorder

\[ m k \ell m \]

Universiteit Utrecht
Involutive monoid on French terms $L^#$

Definition

terms on strings on $\triangleright$-ordered letters

$mk \ell m$

inorder

maxsplit $m > k, \ell$
Involutive monoid on French terms $L^\#$

**Definition**

terms on strings on $\rightarrow$-ordered letters

![Diagram of involutive monoid on French terms](image-url)
Involutive monoid on French terms $L^\#$

**Definition**
terms on strings on $\triangleright$-ordered letters where $\flat \circ \# \text{ identity}$
Involutive monoid on French terms $L^\|$ 

**Definition**

terms on strings on $\succ$-ordered letters where $\blacklozenge \circ \sharp$ identity

\[
\forall \text{ letters } \exists \succ \text{-relating letter in ancestor}
\]

\[
\triangleright \text{-incomparable}
\]

\[
\begin{array}{c}
m & m \\
\varepsilon & k & \ell & \varepsilon \\
\varepsilon & \varepsilon & \varepsilon \\
mk\ell m
\end{array}
\]
Involutive monoid on French terms $L^\dagger$

Definition

terms on strings on $\triangleright$-ordered letters where $\triangleright\circ\dagger$ identity
Involutive monoid on French terms $L^\#$

Definition

terms on **French** strings on $\rightarrow$-ordered letters where $\flat \circ \sharp$ identity operations on $L^\#$ defined via $\hat{L}$, e.g. $t \cdot u = (t^\flat u^\flat)^\#$
A well-founded order on French terms

- (iterative) lexicographic path order based on $\succ$

\[
\begin{array}{c}
\hat{m} \hat{m} \\
\hat{k} \hat{l} \\
\epsilon \epsilon \epsilon
\end{array}
\quad
\begin{array}{c}
\hat{m} \hat{m} \\
\hat{l} \hat{l} \\
\epsilon \epsilon \epsilon
\end{array}
\]
A well-founded order on French terms

- (iterative) lexicographic path order based on >
- lexicographic order on argument places compatible with marks
A well-founded order on French terms

- (iterative) lexicographic path order based on $>$
- lexicographic order on argument places compatible with marks
- signature ordered by $\triangleright = (\triangleright^\text{mul}, \triangleright^\text{area})$ via $\triangleright$
A well-founded order on French strings/terms

- (iterative) lexicographic path order based on $>$
- lexicographic order on argument places compatible with marks
- signature ordered by $\triangleright = (\triangleright_{\text{mul}}) \text{ via } (\triangleright_{\text{area}})$
Properties of $\triangleright_{lpo}$

- head of term $\triangleright$-related to heads of all subterms
Properties of $\triangleright_{lpo}$

- head of term $\triangleright$-related to heads of all subterms
- $\triangleright_{lpo}$ not an ordered monoid: $\hat{k\ell} \triangleright_{lpo} \ell$ but $\hat{k\ell\ell} \not\triangleright_{lpo} \ell\ell$
Properties of $\succ_{lpo}$

- head of term $\succ$-related to heads of all subterms
- $\succ_{lpo}$ not an ordered monoid
- $s \hat{\ell} r \succ_{lpo} s\{\ell\} r$ (in EBNF $\{ \}$ is arbitrary repetition)
Properties of $\triangleright_{lpo}$

- head of term $\triangleright$-related to heads of all subterms
- $\triangleright_{lpo}$ not an ordered monoid
- $s\hat{\ell}r \triangleright_{lpo} s\{\ell>\}r$

**Proof.**

Induction on length $sr$, cases whether $\ell$ is $\triangleright$-maximal in $s\hat{\ell}r$

- **yes** decrease in multiset of head
- **no** induction on substring/term $\hat{\ell}$ is in
Properties of $\succ_{lpo}$

- head of term $\succ$-related to heads of all subterms
- $\succ_{lpo}$ not an ordered monoid
- $s\hat{\ell}r \succ_{lpo} s\{\ell\}r$

Proof.

induction on length $sr$, cases whether $\ell$ is $\succ$-maximal in $s\hat{\ell}r$

- yes decrease in multiset of head
- no induction on substring/term $\hat{\ell}$ is in

- $s\hat{\ell}\hat{m}r \succ_{lpo} s\{\ell\}[\hat{m}][\ell, m>][\hat{\ell}][m>\}r$ ([ ] is option)
Properties of $\succ_{lpo}$

- head of term $\succ$-related to heads of all subterms
- $\succ_{lpo}$ not an ordered monoid
- $s\hat{\ell}r \succ_{lpo} s\{\ell>\}r$

Proof.
induction on length $sr$, cases whether $\ell$ is $\succ$-maximal in $s\hat{\ell}r$

- yes decrease in multiset of head
  - no induction on substring/term $\hat{\ell}$ is in

  - $s\hat{\ell}\hat{m}r \succ_{lpo} s\{\ell>\}[\hat{m}][\ell, m>][\hat{\ell}][\{m>\}r$

Proof.
induction on length $sr$, cases whether $\ell, m$ are $\succ$-maximal in $s\hat{\ell}\hat{m}r$

- both decrease in area of head
  - $\hat{\ell}$ decrease in the substring/term to the right of $\hat{\ell}$
  - $\hat{m}$ decrease in the substring/term to the left of $\hat{m}$

neither induction on substring/term $\hat{\ell}\hat{m}$ is in
Filling in locally decreasing diagram decreases

Theorem
Filling in locally decreasing diagram decreases

Theorem
Filling in locally decreasing diagram decreases

Theorem

Proof.

\[ s \ell m r \succ_{lp} s\{\ell\} [\dot{m}]\{\ell, m\} [\dot{l}] \{m\} r \]
Idea: $\rightarrow$-maximal steps modulo non-$\rightarrow$-maximal steps

case 1: local confluence peak of $\rightarrow$-maximal steps
Idea: $\triangleright$-maximal steps modulo non-$\triangleright$-maximal steps

area decrease
Idea: \(\succ\)-maximal steps modulo non-\(\succ\)-maximal steps

case 2: local coherence peak of \(\succ\)-maximal and non-\(\succ\)-maximal step
Idea: $\triangleright$-maximal steps modulo non-$\triangleright$-maximal steps

decrease in $j$th argument, lexicographically before $i$th
Idea: $\triangleright$-maximal steps modulo non-$\triangleright$-maximal steps

case 3: local modulo peak of non-$\triangleright$-maximal steps
Idea: \(-\text{maximal}\) steps modulo \(\text{non-}(-\text{maximal})\) steps

![Diagram showing the concept of decreasing tiles and involutive proofs.]

decrease in argument both steps are in
\[ \text{\textit{Ipo at work}} \]
Filling in local diagrams

\[ \begin{array}{c}
\text{a} \quad \text{b} \quad \text{c} \\
\downarrow \quad \quad \quad \quad \quad \downarrow \\
\text{m} \quad \text{k} \quad \text{l} \\
\downarrow \quad \quad \quad \quad \quad \downarrow \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{[m, m]} \\
\downarrow \\
\text{[k, ℓ]} \\
\downarrow \\
\text{0} \\
\end{array} \]

\[ \begin{array}{c}
\text{[ ]} \quad \text{[ ]} \\
\downarrow \\
\text{[ ]} \\
\end{array} \]

\[ \begin{array}{c}
\text{[ ]} \\
\downarrow \\
\text{0} \\
\end{array} \]
Filling in local diagrams

\[ \begin{array}{c}
\text{a} \\
\downarrow \ell \\
\text{f} \\
\downarrow m \\
\text{c} \\
\downarrow d \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{b} \\
\downarrow k \\
\text{m} \\
\end{array} \]

\[ \begin{array}{c}
\text{a} \\
\downarrow \ell \\
\text{f} \\
\downarrow m \\
\text{c} \\
\downarrow d \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{b} \\
\downarrow k \\
\text{m} \\
\end{array} \]

\[ \begin{array}{c}
\text{a} \\
\downarrow \ell \\
\text{f} \\
\downarrow m \\
\text{c} \\
\downarrow d \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{b} \\
\downarrow k \\
\text{m} \\
\end{array} \]

\[ \begin{array}{c}
\text{a} \\
\downarrow \ell \\
\text{f} \\
\downarrow m \\
\text{c} \\
\downarrow d \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{b} \\
\downarrow k \\
\text{m} \\
\end{array} \]

\[ \begin{array}{c}
\text{a} \\
\downarrow \ell \\
\text{f} \\
\downarrow m \\
\text{c} \\
\downarrow d \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{b} \\
\downarrow k \\
\text{m} \\
\end{array} \]

\[ \begin{array}{c}
\text{a} \\
\downarrow \ell \\
\text{f} \\
\downarrow m \\
\text{c} \\
\downarrow d \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{b} \\
\downarrow k \\
\text{m} \\
\end{array} \]

\[ \begin{array}{c}
\text{a} \\
\downarrow \ell \\
\text{f} \\
\downarrow m \\
\text{c} \\
\downarrow d \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{b} \\
\downarrow k \\
\text{m} \\
\end{array} \]

\[ \begin{array}{c}
\text{a} \\
\downarrow \ell \\
\text{f} \\
\downarrow m \\
\text{c} \\
\downarrow d \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{b} \\
\downarrow k \\
\text{m} \\
\end{array} \]
Filling in local diagrams
Filling in local diagrams

\[
\begin{array}{c}
& & d \\
& \ell & \\
\ell & \rightarrow & m & \rightarrow & c & \rightarrow & m & \rightarrow & g & \rightarrow & \ell & \rightarrow & e \\
& & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
a & \rightarrow & f & \rightarrow & m & \rightarrow & c & \rightarrow & m & \rightarrow & g & \rightarrow & \ell & \rightarrow & e \\
\end{array}
\]
Filling in local diagrams

\[ [m, m] \]

2 1 3

2 1 1 2

g f
c m m

\[ \ell f \]

\[ a \]

\[ e \]

Applications

Decreasing tiles

Involutive proofs

French strings
Filling in local diagrams

\[
\begin{array}{c}
\text{a} \\
\text{f} \\
\text{g} \\
\text{h} \\
\text{m} \\
\text{i} \\
\text{c} \\
\text{m} \\
\text{e}
\end{array}
\]

\[
\begin{array}{c}
[m, m]_1 \\
\ell \\
\square \\
\square \\
\ell \\
\square \\
\square \\
\square \\
\square \\
\ell \\
\square \\
\square \\
\square \\
\square \\
\ell \\
\square \\
\square \\
\square \\
\square \\
\ell
\end{array}
\]

\[\text{lpo}\]
Filling in local diagrams

\[
\begin{array}{c}
\begin{array}{c}
\text{a} \\
\ell \\
h \\
m \\
i \\
g \\
e
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\begin{array}{c}
[m, m] \\
\ell, \ell \\
[\ell] \\
[]
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\begin{array}{c}
[] \\
[\ell] \\
[]
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\begin{array}{c}
[\ell, \ell] \\
\ell \\
[i, m, m]
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\begin{array}{c}
[] \\
[] \\
[]
\end{array}
\end{array}
\]

35
Filling in local diagrams

\[ \begin{array}{c}
\begin{array}{c}
\text{\textcircled{4}} \quad g \quad \ell \\
\end{array}
\end{array} \]

\begin{align*}
\ell & \quad \ell \quad f \\
\ell & \quad \ell \quad h \\
\ell & \quad m \\
i & \quad m \\
g & \quad \ell \\
e &
\end{align*}

\[ \begin{array}{c}
\begin{array}{c}
[\ell, \ell] \\
_1
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[m, m] \\
_0
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[\ell] \\
_0
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[\ell, \ell] \\
_1
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[m, m] \\
_0
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[\ell] \\
_0
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[\ell] \\
_0
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[\ell] \\
_0
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[\ell] \\
_0
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[\ell] \\
_0
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[\ell] \\
_0
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[\ell] \\
_0
\end{array}
\end{array} \]

\[ \begin{array}{c}
\begin{array}{c}
[\ell] \\
_0
\end{array}
\end{array} \]
Filling in local diagrams

Decreasing tiles
Involutive proofs
French strings
Applications
Filling in local diagrams

Decreasing tiles
Involutive proofs
French strings
Applications
Filling in local diagrams

\[ \begin{align*}
\text{Decreasing tiles} & \quad \text{Involutive proofs} \\
\text{French strings} & \quad \text{Applications}
\end{align*} \]
Filling in local diagrams

Decreasing tiles
Involutive proofs
French strings
Applications

Universiteit Utrecht
Filling in local diagrams

\[
\begin{array}{c}
a \quad m \quad i \\
m \quad k \quad j \\
[0]_0 \quad [k, \ell]_0 \\
[0]_0 \quad [0]_0 \quad [0]_0
\end{array}
\]
Flexibility

Adaptations:

- **monotonic**: by universal quantification over contexts
  \( s \text{ bigger than } r \text{ if } \forall q_1, q_2, q_1 s q_2 \succ_{lpo} q_1 r q_2 \)
Flexibility

Adaptations:

- **monotonic**: by universal quantification over contexts
  \((s \text{ bigger than } r \text{ if } \forall q_1, q_2, q_1sq_2 \triangleright_{lpo} q_1rq_2)\)

- **decidable**: by universal quantification over orders extending
  \((s \text{ bigger than } r \text{ if } \forall \text{ well-orders extending } >, \text{ they are related})\)
Flexibility

Adaptations:

- **monotonic**: by universal quantification over contexts
  
  \( s \) bigger than \( r \) if \( \forall q_1, q_2, q_1 sq_2 \succ_{lpo} q_1 rq_2 \)

- **decidable**: by universal quantification over orders extending
  
  \( s \) bigger than \( r \) if \( \forall \) well-orders extending \( \succ \), they are related

- decreasing diagrams **modulo**: involutive letters \( \dot{l} \), i.e. \( \dot{l}^{-1} = \dot{l} \)
Flexibility

Adaptations:

- **monotonic**: by universal quantification over contexts
  \((s \text{ bigger than } r \text{ if } \forall q_1, q_2, q_1sq_2 \succ_{lpo} q_1rq_2)\)

- **decidable**: by universal quantification over orders extending
  \((s \text{ bigger than } r \text{ if } \forall \text{ well-orders extending } \succ, \text{ they are related})\)

- involutive rewriting \((\varrho : s \rightarrow r \text{ converse of } \varrho^{-1} : s^{-1} \rightarrow r^{-1})\)
Flexibility

Adaptations:

- **monotonic**: by universal quantification over contexts
  
  \( s \) bigger than \( r \) if \( \forall q_1, q_2, q_1 s q_2 \succ_{lpo} q_1 r q_2 \)

- **decidable**: by universal quantification over orders extending
  
  \( s \) bigger than \( r \) if \( \forall \) well-orders extending \( > \), they are
  
  related

- involutive rewriting (\( \varrho: s \to r \) converse of \( \varrho^{-1}: s^{-1} \to r^{-1} \))

- covers all confluence modulo results in Ohlebusch
  
  (either by the previous item, or ordering modulo steps below other steps)
Flexibility

Adaptations:

- **monotonic**: by universal quantification over contexts
  
  \[(s \text{ bigger than } r \text{ if } \forall q_1, q_2, q_1 s q_2 \succ_{lpo} q_1 r q_2)\]

- **decidable**: by universal quantification over orders extending
  
  \[(s \text{ bigger than } r \text{ if } \forall \text{ well-orders extending } >, \text{ they are related})\]

- involutive rewriting \((\varrho : s \rightarrow r \text{ converse of } \varrho^{-1} : s^{-1} \rightarrow r^{-1})\)

- covers all confluence modulo results in Ohlebusch
  
  (either by the previous item, or ordering modulo steps below other steps)

- application to factorisation theorems
  
  (factorisation is commutation with the inverse, RTA 2012, Beniamino Accattoli)
Conclusion

- alternative correctness proof of decreasing diagrams (De Bruijn, vO, Klop, de Vrijer, Bezem, Jouannaud)
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of $\succ$-maximal steps modulo non-$\succ$-maximal steps
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of $\succ$-maximal steps modulo non-$\succ$-maximal steps
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of $\triangleright$-maximal steps modulo non-$\triangleright$-maximal steps
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of $\rightarrow$-maximal steps modulo non-$\rightarrow$-maximal steps
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of $\rightarrow$-maximal steps modulo non-$\rightarrow$-maximal steps
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of $\rightarrow$-maximal steps modulo non-$\rightarrow$-maximal steps
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of >-maximal steps modulo non->-maximal steps
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of $\succ\text{-maximal}$ steps modulo non-$\succ\text{-maximal}$ steps
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of $\succ$-maximal steps modulo non-$\succ$-maximal steps

- Newman’s Lemma (multiset) + Lemma of Hindley–Rosen (area)
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of $\rightarrow$-maximal steps modulo non-$\rightarrow$-maximal steps

- Newman’s Lemma + Lemma of Hindley–Rosen
Conclusion

- alternative correctness proof of decreasing diagrams
- confluence of $\rightarrow$-maximal steps modulo non-$\rightarrow$-maximal steps

- Newman’s Lemma + Lemma of Hindley–Rosen

- flexible
Het kind en ik

Ik zou een dag uit vissen,
ik voelde mij moedeloos.
Ik maakte tussen de lissen
met de hand een wak in het kroos.

Er steeg licht op van beneden
uit de zwarte spiegelgrond.
Ik zag een tuin onbetreden
en een kind dat daar stond.

Het stond aan zijn schrijftafel
te schrijven op een lei.
Het woord onder de griffel
herkende ik, was van mij.

Maar toen heeft het geschreven,
zonder haast en zonder schroom,
al wat ik van mijn leven
nog ooit te schrijven droom.

En telkens als ik even
knikte dat ik het wist,
liet hij het water beven
en het werd uitgewist.

Het kind en ik
Ik zou een dag uit vissen,
ik voelde mij moedeloos.
Ik maakte tussen de lissen
met de hand een wak in het kroos.

Het kind en ik
Ik zou een dag uit vissen,
ik voelde mij moedeloos.
Ik maakte tussen de lissen
met de hand een wak in het kroos.