ALGORITHMS FOR GARSIDE GROUPS

ALGORITHMS FOR GARSIDE GROUPS

Patrick Dehornoy

Laboratoire de Mathématiques Nicolas Oresme, Caen

ALGORITHMS FOR GARSIDE GROUPS

Patrick Dehornoy

Laboratoire de Mathématiques Nicolas Oresme, Caen

- Garside groups = a family of biautomatic groups containing braid groups,
  - not too simple (typically: not abelian),
  - not too complicated (typically: word pb. solvable in quadratic time),

ALGORITHMS FOR GARSIDE GROUPS

Patrick Dehornoy

Laboratoire de Mathématiques Nicolas Oresme, Caen

- Garside groups = a family of biautomatic groups containing braid groups,
  - not too simple (typically: not abelian),
  - not too complicated (typically: word pb. solvable in quadratic time),

$\rightsquigarrow$ natural platforms for group-based cryptography.

ALGORITHMS FOR GARSIDE GROUPS

Patrick Dehornoy

Laboratoire de Mathématiques Nicolas Oresme, Caen

- Garside groups = a family of biautomatic groups containing braid groups,
  - not too simple (typically: not abelian),
  - not too complicated (typically: word pb. solvable in quadratic time),

  $\rightsquigarrow$ natural platforms for group-based cryptography.

- Here, two problems:
  - recognizing a Garside group from a presentation;
  - working with a presented Garside group.

- Definition: A Garside system is a pair (M, Δ), s.t.

- Definition: A Garside system is a pair (M, Δ), s.t.
  - M is a cancellative monoid with no invertible and with lcm's and gcd's,

- Definition: A Garside system is a pair (M, Δ), s.t.
  - M is a cancellative monoid with no invertible and with lcm's and gcd's,
  - Δ is a Garside element in M.

$\mathtt{Div}_L(\Delta) = \mathtt{Div}_R(\Delta)$, this set is finite, and generates M

- Definition: A Garside system is a pair (M, Δ), s.t.
  - M is a cancellative monoid with no invertible and with lcm's and gcd's,
  - Δ is a Garside element in M.

$\mathtt{Div}_L(\Delta) = \mathtt{Div}_R(\Delta)$, this set is finite, and generates M

- By Ore's conditions, a Garside monoid embeds in a group of fractions $\rightsquigarrow$

- Definition: A Garside system is a pair (M, Δ), s.t.
  - M is a cancellative monoid with no invertible and with lcm's and gcd's,
  - Δ is a Garside element in M.

$\mathtt{Div}_L(\Delta) = \mathtt{Div}_R(\Delta)$, this set is finite, and generates M

- By Ore's conditions, a Garside monoid embeds in a group of fractions $\rightsquigarrow$

- Definition: A Garside group is a group that is the group of fractions of (at least one) Garside monoid.

- Definition: A Garside system is a pair (M, Δ), s.t.
  - M is a cancellative monoid with no invertible and with lcm's and gcd's,
  - Δ is a Garside element in M.

$\mathtt{Div}_L(\Delta) = \mathtt{Div}_R(\Delta)$, this set is finite, and generates M

- By Ore's conditions, a Garside monoid embeds in a group of fractions $\rightsquigarrow$

- Definition: A Garside group is a group that is the group of fractions of (at least one) Garside monoid.

- Principle: A Garside group is controlled by the finite lattice $\mathtt{Div}(\Delta)$.

- $(\mathbb{Z}_{>0}, *)$ admits lcm's, but is not Garside: no Δ s.t. $\mathbf{Div}(\Delta)$ generating.

- $(\mathbb{Z}_{>0}, *)$ admits lcm's, but is not Garside: no Δ s.t. $\mathtt{Div}(\Delta)$ generating.

- Artin's braid group B_n (Garside's original example):
$\langle \sigma_1, ..., \sigma_{n-1} ; \sigma_i\sigma_j = \sigma_j\sigma_i$ for $|i-j| \geqslant 2, \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j$ for $|i-j| = 1\rangle$;

- $(\mathbb{Z}_{>0}, *)$ admits lcm's, but is not Garside: no Δ s.t. $\mathtt{Div}(\Delta)$ generating.

- Artin's braid group B_n (Garside's original example):
  $\langle \sigma_1, ..., \sigma_{n-1} ; \sigma_i \sigma_j = \sigma_j \sigma_i$ for $|i-j| \geqslant 2, \sigma_i \sigma_j \sigma_i = \sigma_j \sigma_i \sigma_j$ for $|i-j| = 1 \rangle$;
  - monoid: $B_n^+ := \langle ... \rangle^+$,
  - Garside element: $\Delta_n = \sigma_1 \sigma_2 \sigma_1 \sigma_3 \sigma_2 \sigma_1 ...$;

- $(\mathbb{Z}_{>0}, *)$ admits lcm's, but is not Garside: no Δ s.t. $\mathtt{Div}(\Delta)$ generating.

- Artin's braid group B_n (Garside's original example):
$\langle \sigma_1, ..., \sigma_{n-1}; \sigma_i\sigma_j = \sigma_j\sigma_i$ for $|i-j| \geqslant 2, \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j$ for $|i-j| = 1\rangle$;
  - monoid: $B_n^+ := \langle ... \rangle^+$,
  - Garside element: $\Delta_n = \sigma_1\sigma_2\sigma_1\sigma_3\sigma_2\sigma_1 ...$;
    - $\rightsquigarrow$ lattice $\mathtt{Div}(\Delta_n) \approx (S_n,$, weak order$)$

- $(\mathbb{Z}_{>0}, *)$ admits lcm's, but is not Garside: no Δ s.t. $\mathtt{Div}(\Delta)$ generating.

- Artin's braid group B_n (Garside's original example):
  $\langle \sigma_1, ..., \sigma_{n-1}; \sigma_i\sigma_j = \sigma_j\sigma_i$ for $|i-j| \geqslant 2, \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j$ for $|i-j| = 1\rangle$;
  - monoid: $B_n^+ := \langle ... \rangle^+$,
  - Garside element: $\Delta_n = \sigma_1\sigma_2\sigma_1\sigma_3\sigma_2\sigma_1 ...$;
    $\rightsquigarrow$ lattice $\mathtt{Div}(\Delta_n) \approx (S_n,$, weak order$)$

- Free abelian group of finite type
  $\langle a_1, ..., a_n; a_ia_j = a_ja_i\rangle$;

- $(\mathbb{Z}_{>0}, *)$ admits lcm's, but is not Garside: no Δ s.t. $\mathtt{Div}(\Delta)$ generating.

- Artin's braid group B_n (Garside's original example):
$\langle \sigma_1, ..., \sigma_{n-1}; \sigma_i\sigma_j = \sigma_j\sigma_i$ for $|i-j| \geqslant 2, \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j$ for $|i-j| = 1\rangle$;
  - monoid: $B_n^+ := \langle ... \rangle^+$,
  - Garside element: $\Delta_n = \sigma_1\sigma_2\sigma_1\sigma_3\sigma_2\sigma_1 ...$;
    - $\rightsquigarrow$ lattice $\mathtt{Div}(\Delta_n) \approx (S_n,$, weak order$)$

Δ_4

1

- Free abelian group of finite type
$\langle a_1, ..., a_n; a_ia_j = a_ja_i\rangle$;
  - monoid: $\langle a_1, ..., a_n; a_ia_j = a_ja_i\rangle^+$;
  - Garside element: $\Delta = a_1 ... a_n$;

- $(\mathbb{Z}_{>0}, *)$ admits lcm's, but is not Garside: no Δ s.t. $\mathtt{Div}(\Delta)$ generating.

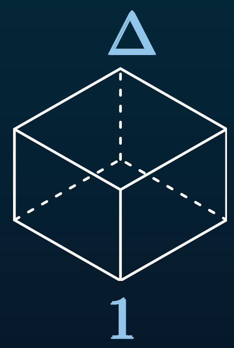
- Artin's braid group B_n (Garside's original example):
$\langle \sigma_1, ..., \sigma_{n-1}; \sigma_i\sigma_j = \sigma_j\sigma_i$ for $|i-j| \geqslant 2, \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j$ for $|i-j| = 1\rangle$;
  - monoid: $B_n^+ := \langle ... \rangle^+$,
  - Garside element: $\Delta_n = \sigma_1\sigma_2\sigma_1\sigma_3\sigma_2\sigma_1 ...$;
    $\rightsquigarrow$ lattice $\mathtt{Div}(\Delta_n) \approx (S_n,$, weak order$)$

- Free abelian group of finite type
$\langle a_1, ..., a_n; a_ia_j = a_ja_i\rangle$;
  - monoid: $\langle a_1, ..., a_n; a_ia_j = a_ja_i\rangle^+$;
  - Garside element: $\Delta = a_1 ... a_n$;
    $\rightsquigarrow$ lattice $\mathtt{Div}(\Delta) \approx$ cube

- $(\mathbb{Z}_{>0}, *)$ admits lcm's, but is not Garside: no Δ s.t. $\mathtt{Div}(\Delta)$ generating.

- Artin's braid group B_n (Garside's original example):
  $\langle \sigma_1, ..., \sigma_{n-1} ; \sigma_i\sigma_j = \sigma_j\sigma_i$ for $|i-j| \geqslant 2, \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j$ for $|i-j| = 1\rangle$;
  - monoid: $B_n^+ := \langle ... \rangle^+$,
  - Garside element: $\Delta_n = \sigma_1\sigma_2\sigma_1\sigma_3\sigma_2\sigma_1 ...$;
    $\rightsquigarrow$ lattice $\mathtt{Div}(\Delta_n) \approx (S_n,$, weak order$)$

- Free abelian group of finite type
  $\langle a_1, ..., a_n ; a_ia_j = a_ja_i \rangle$;
  - monoid: $\langle a_1, ..., a_n ; a_ia_j = a_ja_i \rangle^+$;
  - Garside element: $\Delta = a_1 ... a_n$;
    $\rightsquigarrow$ lattice $\mathtt{Div}(\Delta) \approx$ cube

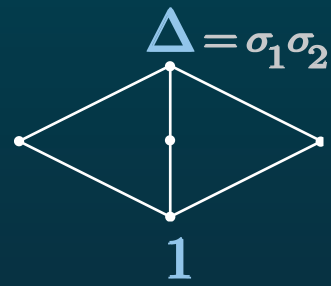


- More generally: spherical Artin–Tits groups
  $\rightsquigarrow$ lattice = weak order on the associated Coxeter group

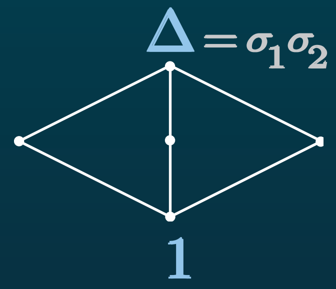
- Also: torus knots groups $\langle a, b, c, \ldots; a^p = b^q = c^r = \ldots \rangle$...

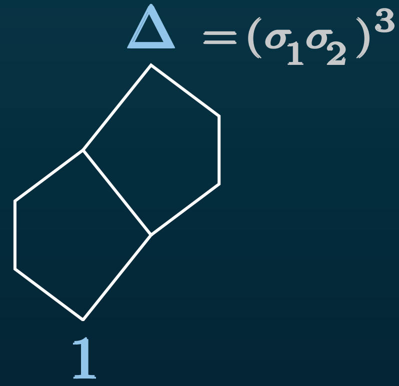
- Also: torus knots groups $\langle a, b, c, ...; a^p = b^q = c^r = ...\rangle$...
- Dual Garside structure on B_n (Birman-Ko-Lee, Bessis...):
  $\rightsquigarrow$ same group, different monoid;

- Also: torus knots groups $\langle a, b, c, \ldots; a^p = b^q = c^r = \ldots\rangle$...
- Dual Garside structure on B_n (Birman-Ko-Lee, Bessis...):
  - $\rightsquigarrow$ same group, different monoid;
  - $\rightsquigarrow$ case of B_3: $\langle a, b, c; ab = bc = ca\rangle$



- Also: torus knots groups $\langle a, b, c, ...; a^p = b^q = c^r = ...\rangle$...
- Dual Garside structure on B_n (Birman-Ko-Lee, Bessis...):
  - ⇝ same group, different monoid;
  - ⇝ case of B_3: $\langle a, b, c; ab = bc = ca\rangle$





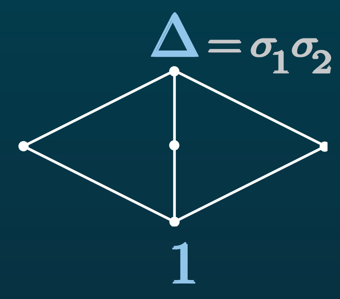
- Also, always for B_3: $\langle a, b; aba = b^2\rangle$:

EXAMPLES (2)

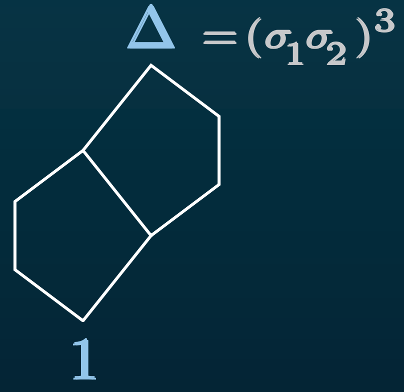
- Also: torus knots groups $\langle a, b, c, \ldots; a^p = b^q = c^r = \ldots\rangle$...
- Dual Garside structure on B_n (Birman-Ko-Lee, Bessis...):

  ⇝ same group, different monoid;

  ⇝ case of B_3: $\langle a, b, c; ab = bc = ca\rangle$



- Also, always for B_3: $\langle a, b; aba = b^2\rangle$:



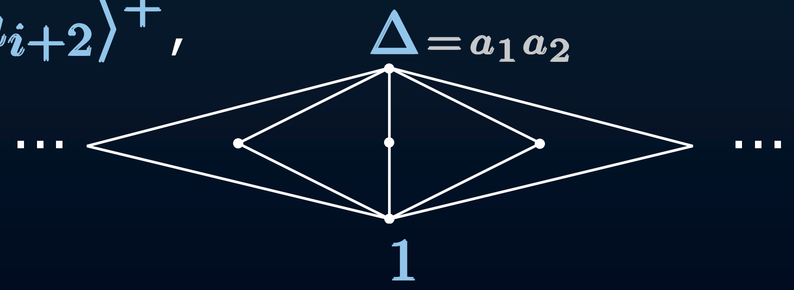
like Garside but with $\mathbf{Div}(\Delta)$ finite height only (not necessarily finite)

↓

- Free groups are quasi-Garside (Bessis, Brady-Crisp-Kaul-McCammond)

F_2: - monoid $\langle \ldots a_{-1}, a_0, a_1, \ldots; a_i a_{i+1} = a_{i+1} a_{i+2}\rangle^+$,

- quasi-Garside element $\Delta = a_i a_{i+1}$:



- Problem #1: Recognize a Garside group from a presentation;

- Problem #1: Recognize a Garside group from a presentation;
- Problem #2: Compute in a Garside group given by a presentation.

- Problem #1: Recognize a Garside group from a presentation;
- Problem #2: Compute in a Garside group given by a presentation.

- Assume that M is a Garside monoid generated by X, and R contains one relation $sv = tu$ representing $\mathbf{lcm}(s, t)$ for all s, t in X. Then (X, R) is a presentation of M (and of the group of fractions of M).

- Problem #1: Recognize a Garside group from a presentation;
- Problem #2: Compute in a Garside group given by a presentation.

- Assume that M is a Garside monoid generated by X, and
  R contains one relation $sv = tu$ representing $\mathrm{lcm}(s, t)$ for all s, t in X.
  Then (X, R) is a presentation of M (and of the group of fractions of M).

- For such a presentation by construction:

- all relations of the form $u = v$ with u, v nonempty positive words (no s^{-1});
- no relation $su = sv$ with $u \neq v$;
- at most one relation $su = tv$ for all s, t.

- Problem #1: Recognize a Garside group from a presentation;
- Problem #2: Compute in a Garside group given by a presentation.

- Assume that M is a Garside monoid generated by X, and
  R contains one relation $sv = tu$ representing $\mathbf{lcm}(s, t)$ for all s, t in X.
  Then (X, R) is a presentation of M (and of the group of fractions of M).

- For such a presentation by construction:

- all relations of the form $u = v$ with u, v nonempty positive words (no s^{-1});
- no relation $su = sv$ with $u \neq v$;
- at most one relation $su = tv$ for all s, t.

$\rightsquigarrow$ a "complemented" presentation.

- Problem #1: Recognize a Garside group from a presentation;
- Problem #2: Compute in a Garside group given by a presentation.

- Assume that M is a Garside monoid generated by X, and
  R contains one relation $sv = tu$ representing $\mathrm{lcm}(s, t)$ for all s, t in X.
  Then (X, R) is a presentation of M (and of the group of fractions of M).

- For such a presentation by construction:

- all relations of the form $u = v$ with u, v nonempty positive words (no s^{-1});
- no relation $su = sv$ with $u \neq v$;
- at most one relation $su = tv$ for all s, t.

$\rightsquigarrow$ a "complemented" presentation.

$\rightsquigarrow$ wlog:

- Pb #1: Recognize a Garside monoid from a complemented presentation;
- Pb #2: Compute in a Garside group given by a complemented presentation.

- Word reversing: a syntactic method relevant for semigroup presentations

that computes lcm's in good cases (?)

- Word reversing: a syntactic method relevant for semigroup presentations
  that computes lcm's in good cases (?)

words “for the group”

- Defin.: For (S, R) a semigroup presentation, and w, w' words on $S \cup S^{-1}$,

- Word reversing: a syntactic method relevant for semigroup presentations

  that computes lcm's in good cases (?)

words "for the group"

- Defin.: For (S, R) a semigroup presentation, and w, w' words on $S \cup S^{-1}$,

  $w \curvearrowright w'$ ("w reverses to w'"), if w' obtained from w by (iteratively)

  - deleting some $s^{-1}s$, or

- Word reversing: a syntactic method relevant for semigroup presentations

  that computes lcm's in good cases (?)

  words "for the group"

- Defin.: For (S, R) a semigroup presentation, and w, w' words on $S \cup S^{-1}$,

  $w \curvearrowright w'$ ("w reverses to w'"), if w' obtained from w by (iteratively)

  - deleting some $s^{-1}s$, or
  - replacing some $s^{-1}t$ with vu^{-1} s.t. $sv = tu \in R$.

- Word reversing: a syntactic method relevant for semigroup presentations

  that computes lcm's in good cases (?)

words "for the group"

- Defin.: For (S, R) a semigroup presentation, and w, w' words on $S \cup S^{-1}$,

  $w \curvearrowright w'$ ("w reverses to w'"), if w' obtained from w by (iteratively)

  - deleting some $s^{-1}s$, or
  - replacing some $s^{-1}t$ with vu^{-1} s.t. $sv = tu \in R$.

- Remark 1: Deleting $s^{-1}s$ is reversing w.r.t. $s = s$;

- Word reversing: a syntactic method relevant for semigroup presentations

that computes lcm's in good cases (?)

words "for the group"

- Defin.: For (S, R) a semigroup presentation, and w, w' words on $S \cup S^{-1}$,

$w \curvearrowright w'$ ("w reverses to w'"), if w' obtained from w by (iteratively)

  - deleting some $s^{-1}s$, or
  - replacing some $s^{-1}t$ with vu^{-1} s.t. $sv = tu \in R$.

- Remark 1: Deleting $s^{-1}s$ is reversing w.r.t. $s = s$;
- Remark 2: Deleting ss^{-1} is not legal reversing.

- Word reversing: a syntactic method relevant for semigroup presentations that computes lcm's in good cases (?)

words "for the group"

- Defin.: For (S, R) a semigroup presentation, and w, w' words on $S \cup S^{-1}$,
  $w \curvearrowright w'$ ("w reverses to w'"), if w' obtained from w by (iteratively)
  - deleting some $s^{-1}s$, or
  - replacing some $s^{-1}t$ with vu^{-1} s.t. $sv = tu \in R$.

- Remark 1: Deleting $s^{-1}s$ is reversing w.r.t. $s = s$;
- Remark 2: Deleting ss^{-1} is not legal reversing.
- Remark 3: $w \curvearrowright w'$ implies $w \equiv w'$.

represents the same element in the group $\langle S; R \rangle$

- Reversing = replacing a $-+$ subword with a $+-$ subword.

- Reversing = replacing a $-+$ subword with a $+-$ subword.
- Example: $S := \{\mathbf{a}, \mathbf{b}\}, R := \{\mathbf{aba} = \mathbf{bb}\}$

$\mathbf{a}^{-1}\mathbf{baa}$

- Reversing = replacing a $-+$ subword with a $+-$ subword.
- Example: $S := \{\mathbf{a}, \mathbf{b}\}, R := \{\mathbf{aba} = \mathbf{bb}\}$

$\mathbf{a}^{-1}\mathbf{baa} \curvearrowright \mathbf{bab}^{-1}\mathbf{aa}$

- Reversing = replacing a $-+$ subword with a $+-$ subword.
- Example: $S := \{\mathtt{a}, \mathtt{b}\}, R := \{\mathtt{aba} = \mathtt{bb}\}$

$\mathtt{a}^{-1}\mathtt{baa} \curvearrowright \mathtt{bab}^{-1}\mathtt{aa} \curvearrowright \mathtt{baba}^{-1}\mathtt{b}^{-1}\mathtt{a}$

- Reversing = replacing a $-+$ subword with a $+-$ subword.

- Example: $S := \{\mathtt{a}, \mathtt{b}\}, R := \{\mathtt{aba} = \mathtt{bb}\}$

$$\mathtt{a}^{-1}\mathtt{baa} \curvearrowright \mathtt{bab}^{-1}\mathtt{aa} \curvearrowright \mathtt{baba}^{-1}\mathtt{b}^{-1}\mathtt{a} \curvearrowright \mathtt{bab}\ \mathtt{a}^{-1}\mathtt{ba}^{-1}\mathtt{b}^{-1}$$

- Reversing = replacing a $-+$ subword with a $+-$ subword.

- Example: $S := \{\mathtt{a}, \mathtt{b}\}, R := \{\mathtt{aba} = \mathtt{bb}\}$

$$\mathtt{a}^{-1}\mathtt{baa} \curvearrowright \mathtt{bab}^{-1}\mathtt{aa} \curvearrowright \mathtt{baba}^{-1}\mathtt{b}^{-1}\mathtt{a} \curvearrowright \mathtt{bab\,a}^{-1}\mathtt{ba}^{-1}\mathtt{b}^{-1} \curvearrowright \mathtt{babbab}^{-1}\mathtt{a}^{-1}\mathtt{b}^{-1}$$

- Reversing = replacing a $-+$ subword with a $+-$ subword.

- Example: $S := \{\mathtt{a}, \mathtt{b}\}, R := \{\mathtt{aba} = \mathtt{bb}\}$

$\mathtt{a}^{-1}\mathtt{baa} \curvearrowright \mathtt{bab}^{-1}\mathtt{aa} \curvearrowright \mathtt{baba}^{-1}\mathtt{b}^{-1}\mathtt{a} \curvearrowright \mathtt{bab}\,\mathtt{a}^{-1}\mathtt{ba}^{-1}\mathtt{b}^{-1} \curvearrowright \mathtt{babbab}^{-1}\mathtt{a}^{-1}\mathtt{b}^{-1}$

... a positive-negative word: cannot be reversed anymore.

- Reversing = replacing a $-+$ subword with a $+-$ subword.

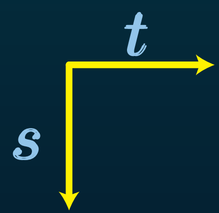
- Example: $S := \{\mathtt{a}, \mathtt{b}\}, R := \{\mathtt{aba} = \mathtt{bb}\}$

$$\mathtt{a}^{-1}\mathtt{baa} \curvearrowright \mathtt{bab}^{-1}\mathtt{aa} \curvearrowright \mathtt{baba}^{-1}\mathtt{b}^{-1}\mathtt{a} \curvearrowright \mathtt{bab}\,\mathtt{a}^{-1}\mathtt{ba}^{-1}\mathtt{b}^{-1} \curvearrowright \mathtt{babbab}^{-1}\mathtt{a}^{-1}\mathtt{b}^{-1}$$

... a positive-negative word: cannot be reversed anymore.

- Reversing = constructing a van Kampen diagram from the source vertices:

$s^{-1}t \curvearrowright vu^{-1} \rightsquigarrow$ replacing



- Reversing = replacing a $-+$ subword with a $+-$ subword.

- Example: $S := \{\mathbf{a}, \mathbf{b}\}, R := \{\mathbf{aba} = \mathbf{bb}\}$

$$\mathbf{a}^{-1}\mathbf{baa} \curvearrowright \mathbf{bab}^{-1}\mathbf{aa} \curvearrowright \mathbf{baba}^{-1}\mathbf{b}^{-1}\mathbf{a} \curvearrowright \mathbf{bab}\,\mathbf{a}^{-1}\mathbf{ba}^{-1}\mathbf{b}^{-1} \curvearrowright \mathbf{babbab}^{-1}\mathbf{a}^{-1}\mathbf{b}^{-1}$$

... a positive-negative word: cannot be reversed anymore.

- Reversing = constructing a van Kampen diagram from the source vertices:

$s^{-1}t \curvearrowright vu^{-1} \rightsquigarrow$ replacing
$$\begin{tikzcd} \cdot \arrow[r, "t"] \arrow[d, "s"'] & \cdot \\ \cdot & \end{tikzcd}$$
with
$$\begin{tikzcd} \cdot \arrow[r, dashed, "t"] \arrow[d, dashed, "s"'] & \cdot \arrow[d, "u"] \\ \cdot \arrow[r, "v"'] & \cdot \end{tikzcd}$$

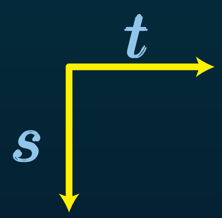
- Reversing = replacing a $-+$ subword with a $+-$ subword.

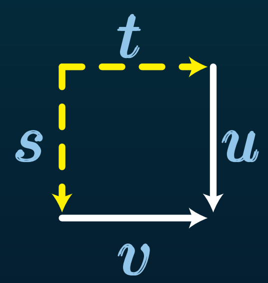
- Example: $S := \{\mathtt{a}, \mathtt{b}\}, R := \{\mathtt{aba} = \mathtt{bb}\}$

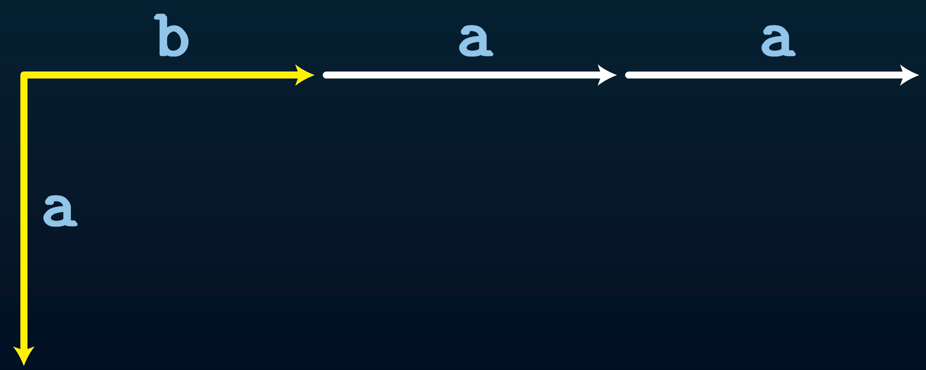
$$\mathtt{a}^{-1}\mathtt{baa} \curvearrowright \mathtt{bab}^{-1}\mathtt{aa} \curvearrowright \mathtt{baba}^{-1}\mathtt{b}^{-1}\mathtt{a} \curvearrowright \mathtt{bab\,a}^{-1}\mathtt{ba}^{-1}\mathtt{b}^{-1} \curvearrowright \mathtt{babbab}^{-1}\mathtt{a}^{-1}\mathtt{b}^{-1}$$

... a positive-negative word: cannot be reversed anymore.

- Reversing = constructing a van Kampen diagram from the source vertices:

$s^{-1}t \curvearrowright vu^{-1} \rightsquigarrow$ replacing 

with 



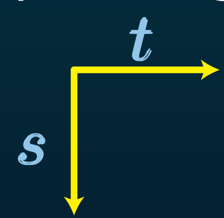
- Reversing = replacing a $-+$ subword with a $+-$ subword.

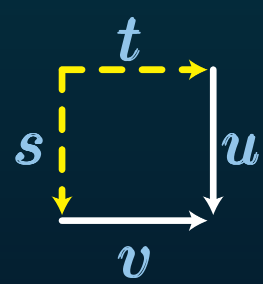
- Example: $S := \{\mathtt{a}, \mathtt{b}\}, R := \{\mathtt{aba} = \mathtt{bb}\}$

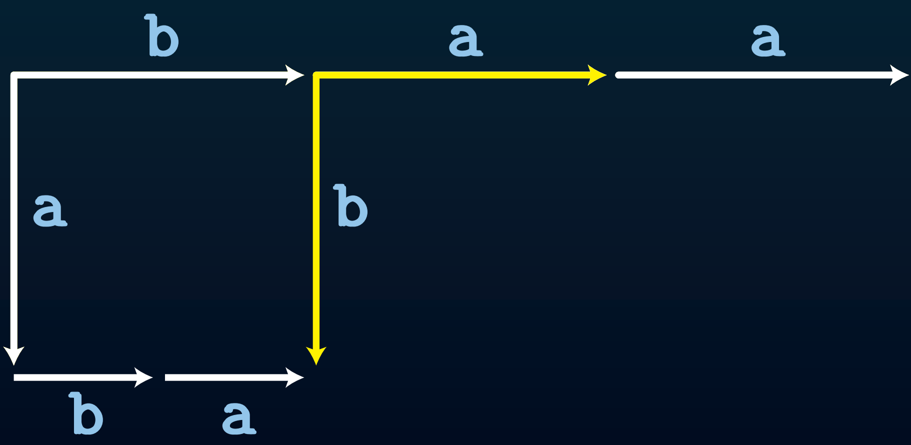
$$\mathtt{a}^{-1}\mathtt{baa} \curvearrowright \mathtt{bab}^{-1}\mathtt{aa} \curvearrowright \mathtt{baba}^{-1}\mathtt{b}^{-1}\mathtt{a} \curvearrowright \mathtt{bab}\;\mathtt{a}^{-1}\mathtt{ba}^{-1}\mathtt{b}^{-1} \curvearrowright \mathtt{babbab}^{-1}\mathtt{a}^{-1}\mathtt{b}^{-1}$$

... a positive-negative word: cannot be reversed anymore.

- Reversing = constructing a van Kampen diagram from the source vertices:

$s^{-1}t \curvearrowright vu^{-1} \rightsquigarrow$ replacing 

with 



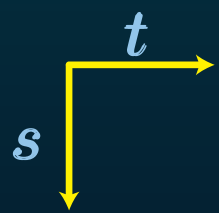
- Reversing = replacing a $-+$ subword with a $+-$ subword.

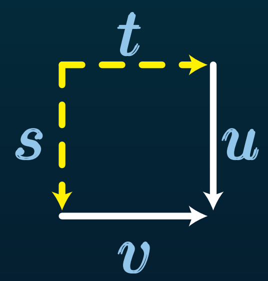
- Example: $S := \{\mathtt{a}, \mathtt{b}\}, R := \{\mathtt{aba} = \mathtt{bb}\}$

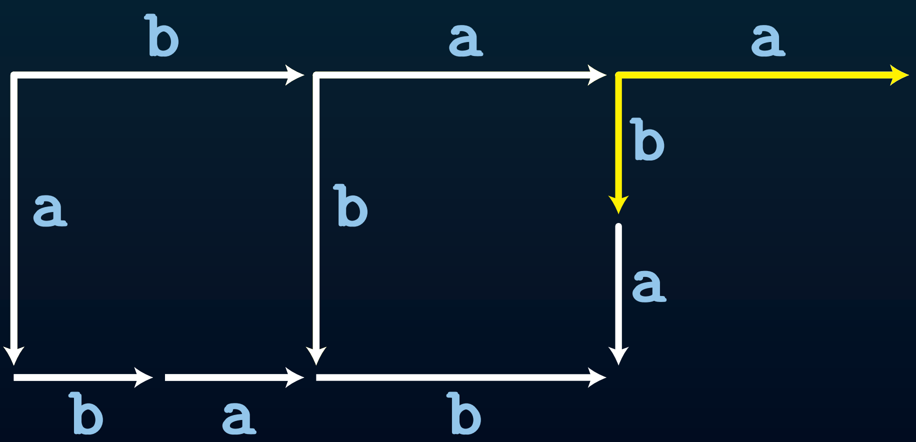
$$\mathtt{a}^{-1}\mathtt{baa} \curvearrowright \mathtt{bab}^{-1}\mathtt{aa} \curvearrowright \mathtt{baba}^{-1}\mathtt{b}^{-1}\mathtt{a} \curvearrowright \mathtt{bab}\,\mathtt{a}^{-1}\mathtt{ba}^{-1}\mathtt{b}^{-1} \curvearrowright \mathtt{babbab}^{-1}\mathtt{a}^{-1}\mathtt{b}^{-1}$$

... a positive-negative word: cannot be reversed anymore.

- Reversing = constructing a van Kampen diagram from the source vertices:

$s^{-1}t \curvearrowright vu^{-1} \rightsquigarrow$ replacing 

with 



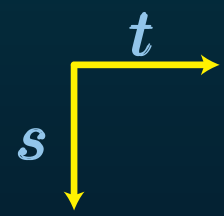
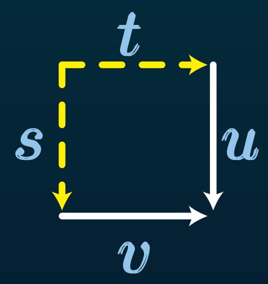
- Reversing = replacing a $-+$ subword with a $+-$ subword.

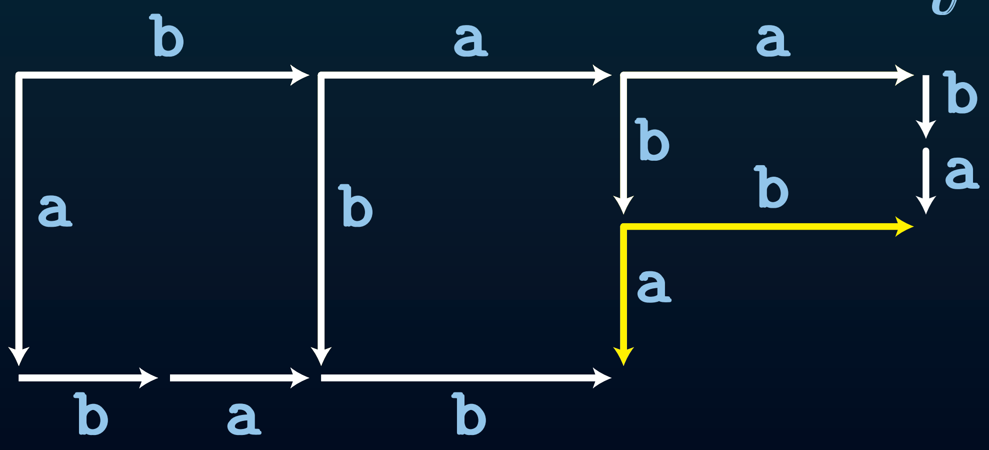
- Example: $S := \{\mathtt{a}, \mathtt{b}\}, R := \{\mathtt{aba} = \mathtt{bb}\}$

$$\mathtt{a}^{-1}\mathtt{baa} \curvearrowright \mathtt{bab}^{-1}\mathtt{aa} \curvearrowright \mathtt{baba}^{-1}\mathtt{b}^{-1}\mathtt{a} \curvearrowright \mathtt{bab\,a}^{-1}\mathtt{ba}^{-1}\mathtt{b}^{-1} \curvearrowright \mathtt{babbab}^{-1}\mathtt{a}^{-1}\mathtt{b}^{-1}$$

... a positive-negative word: cannot be reversed anymore.

- Reversing = constructing a van Kampen diagram from the source vertices:

$s^{-1}t \curvearrowright vu^{-1} \rightsquigarrow$ replacing  with 



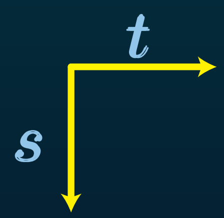
- Reversing = replacing a $-+$ subword with a $+-$ subword.

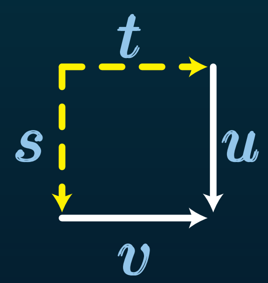
- Example: $S := \{\mathtt{a}, \mathtt{b}\}, R := \{\mathtt{aba} = \mathtt{bb}\}$

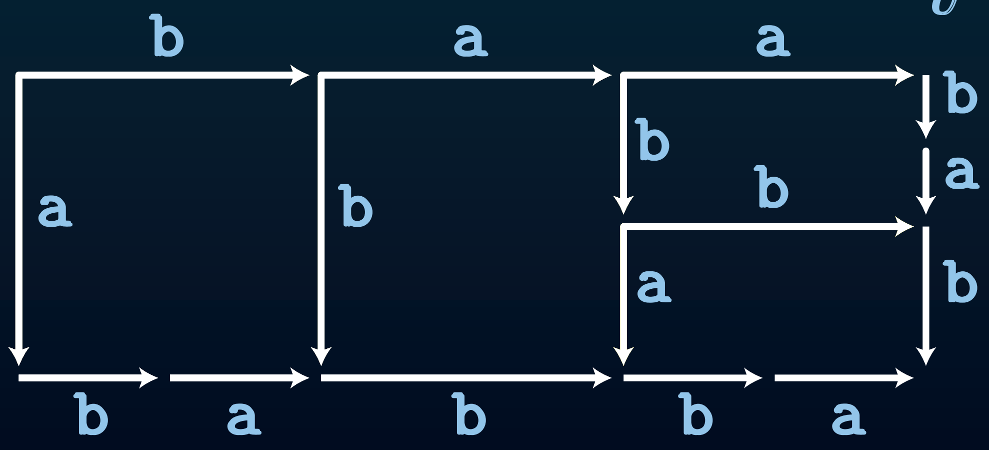
$$\mathtt{a}^{-1}\mathtt{baa} \curvearrowright \mathtt{bab}^{-1}\mathtt{aa} \curvearrowright \mathtt{baba}^{-1}\mathtt{b}^{-1}\mathtt{a} \curvearrowright \mathtt{bab}\ \mathtt{a}^{-1}\mathtt{ba}^{-1}\mathtt{b}^{-1} \curvearrowright \mathtt{babbab}^{-1}\mathtt{a}^{-1}\mathtt{b}^{-1}$$

... a positive-negative word: cannot be reversed anymore.

- Reversing = constructing a van Kampen diagram from the source vertices:

$s^{-1}t \curvearrowright vu^{-1} \rightsquigarrow$ replacing 

with 



- What can one deduce from $w \curvearrowright w'$?

- What can one deduce from $w \curvearrowright w'$?

- Not much: if u, v are positive, $u^{-1}v \curvearrowright \underbrace{\varepsilon}_{\text{the empty word}}$ implies $u \equiv v$, and even $u \underbrace{\equiv^+}_{\text{represent the same element in the monoid}} v$.

- What can one deduce from $w \curvearrowright w'$?

- Not much: if u, v are positive, $u^{-1}v \curvearrowright \underset{\text{the empty word}}{\varepsilon}$ implies $u \equiv v$, and even $u \underset{\text{represent the same element in the monoid}}{\equiv^+} v$.

- The good case: when the converse holds ("reversing detects equivalence").

- What can one deduce from $w \curvearrowright w'$?

the empty word

- Not much: if u, v are positive, $u^{-1}v \curvearrowright \varepsilon$ implies $u \equiv v$, and even $u \equiv^{+} v$.

represent the same element in the monoid

- The good case: when the converse holds ("reversing detects equivalence").

- Definition: The presentation (S, R) is complete for reversing
if $u \equiv^{+} v$ implies (hence, is equivalent to) $u^{-1}v \curvearrowright \varepsilon$.

- What can one deduce from $w \curvearrowright w'$?

the empty word

- Not much: if u, v are positive, $u^{-1}v \curvearrowright \varepsilon$ implies $u \equiv v$, and even $u \equiv^+ v$.

represent the same element in the monoid

- The good case: when the converse holds ("reversing detects equivalence").

- Definition: The presentation (S, R) is complete for reversing
  if $u \equiv^+ v$ implies (hence, is equivalent to) $u^{-1}v \curvearrowright \varepsilon$.

$\rightsquigarrow$ Criterion for recognizing completeness?

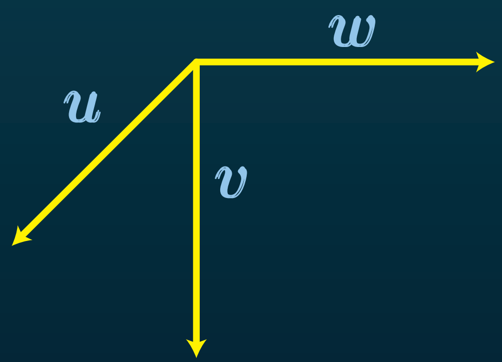
- Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

- Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

$$u^{-1}vv^{-1}w \curvearrowright v'u'^{-1} \text{ implies } (uv')^{-1}(vu') \curvearrowright \varepsilon.$$

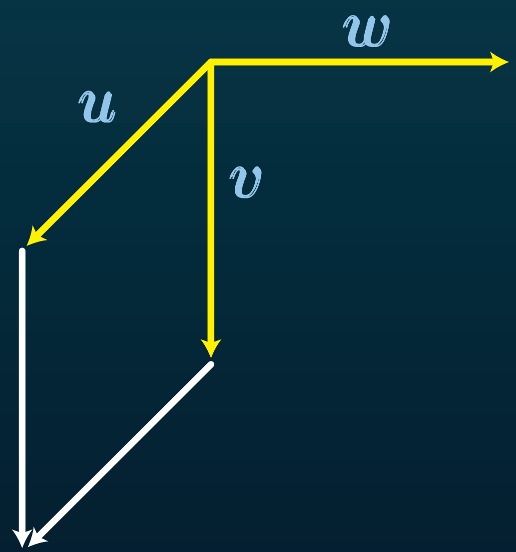
• Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

$$u^{-1}vv^{-1}w \curvearrowright v'u'^{-1} \text{ implies } (uv')^{-1}(vu') \curvearrowright \varepsilon.$$



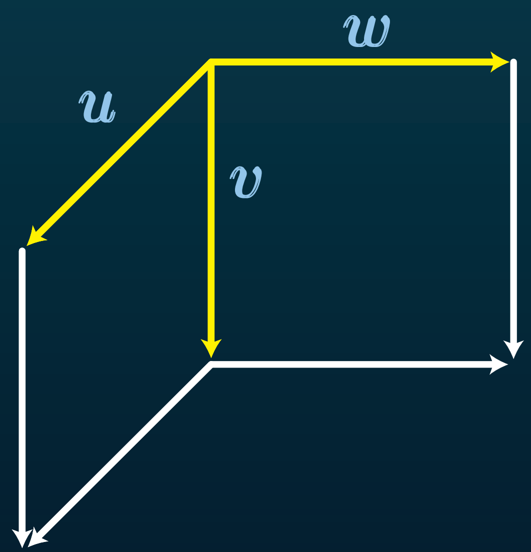
- Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

$$u^{-1}vv^{-1}w \curvearrowright v'u'^{-1} \text{ implies } (uv')^{-1}(vu') \curvearrowright \varepsilon.$$



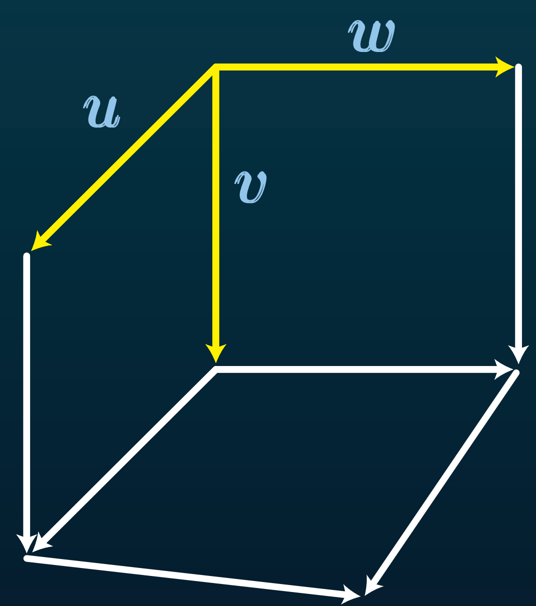
- Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

$$u^{-1}vv^{-1}w \curvearrowright v'u'^{-1} \text{ implies } (uv')^{-1}(vu') \curvearrowright \varepsilon.$$



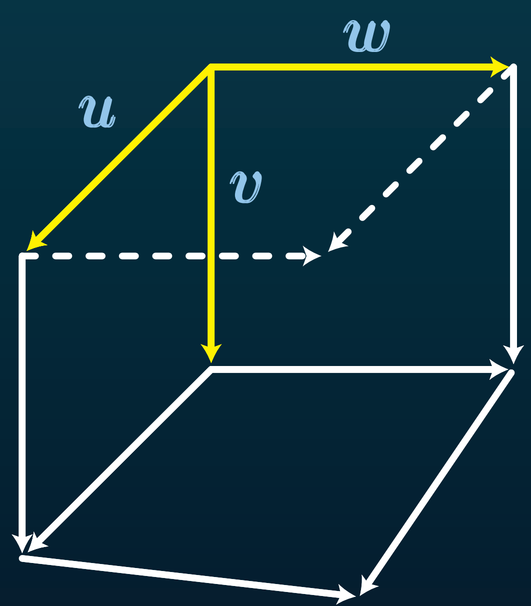
- Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

$$u^{-1}vv^{-1}w \curvearrowright v'u'^{-1} \text{ implies } (uv')^{-1}(vu') \curvearrowright \varepsilon.$$



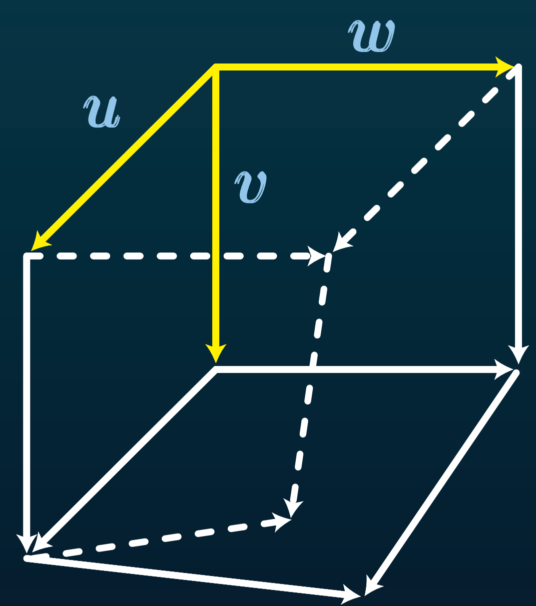
• Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

$$u^{-1}vv^{-1}w \curvearrowright v'u'^{-1} \text{ implies } (uv')^{-1}(vu') \curvearrowright \varepsilon.$$



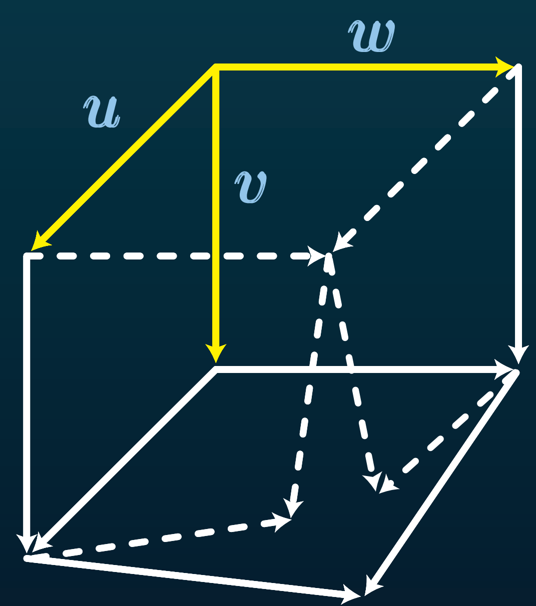
• Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

$$u^{-1}vv^{-1}w \curvearrowright v'u'^{-1} \text{ implies } (uv')^{-1}(vu') \curvearrowright \varepsilon.$$



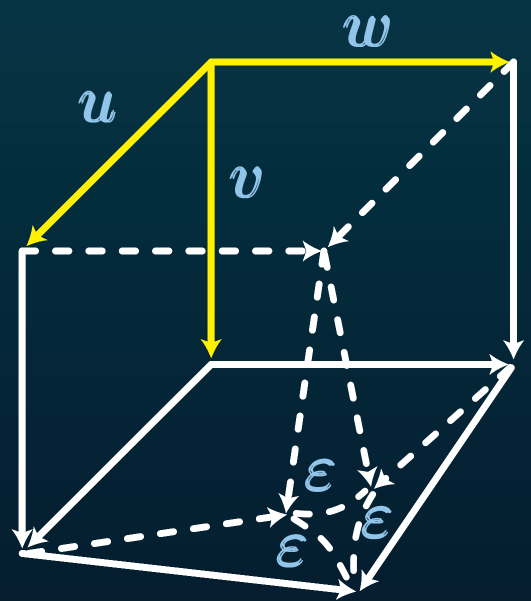
- Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

$$u^{-1}vv^{-1}w \curvearrowright v'u'^{-1} \text{ implies } (uv')^{-1}(vu') \curvearrowright \varepsilon.$$



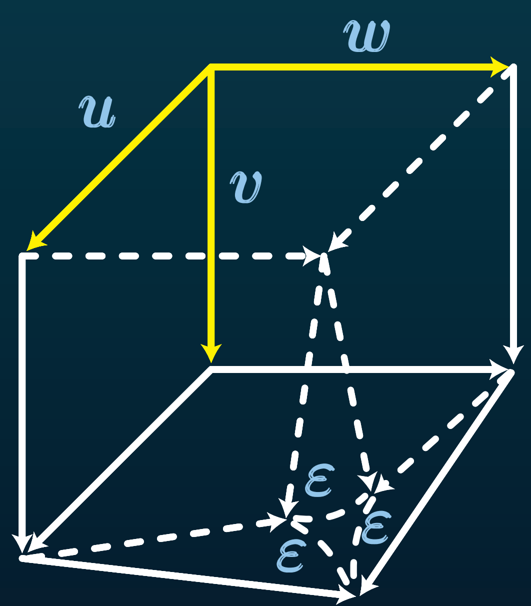
- Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

$$u^{-1}vv^{-1}w \curvearrowright v'u'^{-1} \text{ implies } (uv')^{-1}(vu') \curvearrowright \varepsilon.$$



- Definition: For (S, R) a semigroup presentation, and X set of words on S, say that the cube condition holds on X if, for all u, v, w in X,

$$u^{-1}vv^{-1}w \curvearrowright v'u'^{-1} \text{ implies } (uv')^{-1}(vu') \curvearrowright \varepsilon.$$



- Fact: A semigroup presentation (S, R) is complete for reversing iff the cube condition holds on S^* (***i.e.***, for all words).

- Let $(S; R)$ be a semigroup presentation:

- Let $(S; R)$ be a semigroup presentation:

Criterion 1: If the relations of R preserve some pseudo-length,

$$\lambda : S^* \to \mathbb{N} \text{ s.t. } \lambda(uv) \geqslant \lambda(u) + \lambda(v) \text{ and } \lambda(s) \geqslant 1 \text{ for } s \in S$$

and the cube condition holds on S, then (S, R) is complete for reversing.

- Let $(S;R)$ be a semigroup presentation:

Criterion 1: If the relations of R preserve some pseudo-length,

$\lambda : S^* \to \mathbb{N}$ s.t. $\lambda(uv) \geqslant \lambda(u) + \lambda(v)$ and $\lambda(s) \geqslant 1$ for $s \in S$

and the cube condition holds on S, then (S, R) is complete for reversing.

Criterion 2: If there exists $\widehat{S} \supseteq S$ closed under reversing,

if $u, v \in \widehat{S}$ and $u^{-1}v \curvearrowright v'u'^{-1}$, then $u', v' \in \widehat{S}$

and the cube condition holds on $\widehat{S}$, then (S, R) is complete for reversing.

- Let $(S;R)$ be a semigroup presentation:

Criterion 1: If the relations of R preserve some pseudo-length,

$\uparrow$

$\lambda : S^* \to \mathbb{N}$ s.t. $\lambda(uv) \geqslant \lambda(u) + \lambda(v)$ and $\lambda(s) \geqslant 1$ for $s \in S$

and the cube condition holds on S, then (S,R) is complete for reversing.

Criterion 2: If there exists $\widehat{S} \supseteq S$ closed under reversing,

$\uparrow$

if $u, v \in \widehat{S}$ and $u^{-1}v \curvearrowright v'u'^{-1}$, then $u', v' \in \widehat{S}$

and the cube condition holds on $\widehat{S}$, then (S,R) is complete for reversing.

- Example: $S := \{\mathtt{a}, \mathtt{b}\}$, $R := \{\mathtt{aba} = \mathtt{bb}\}$.

- Let $(S;R)$ be a semigroup presentation:

Criterion 1: If the relations of R preserve some pseudo-length,

$$\lambda : S^* \to \mathbb{N} \text{ s.t. } \lambda(uv) \geqslant \lambda(u) + \lambda(v) \text{ and } \lambda(s) \geqslant 1 \text{ for } s \in S$$

and the cube condition holds on S, then (S,R) is complete for reversing.

Criterion 2: If there exists $\widehat{S} \supseteq S$ closed under reversing,

$$\text{if } u, v \in \widehat{S} \text{ and } u^{-1}v \curvearrowright v'u'^{-1}, \text{ then } u', v' \in \widehat{S}$$

and the cube condition holds on $\widehat{S}$, then (S,R) is complete for reversing.

- Example: $S := \{\mathtt{a}, \mathtt{b}\}$, $R := \{\mathtt{aba} = \mathtt{bb}\}$.

  For Criterion 1: $\lambda(\mathtt{a}) := 1$, $\lambda(\mathtt{b}) := 2$;

- Let $(S;R)$ be a semigroup presentation:

Criterion 1: If the relations of R preserve some pseudo-length,

$$\lambda : S^* \to \mathbb{N} \text{ s.t. } \lambda(uv) \geqslant \lambda(u) + \lambda(v) \text{ and } \lambda(s) \geqslant 1 \text{ for } s \in S$$

and the cube condition holds on S, then (S,R) is complete for reversing.

Criterion 2: If there exists $\widehat{S} \supseteq S$ closed under reversing,

$$\text{if } u, v \in \widehat{S} \text{ and } u^{-1}v \curvearrowright v'u'^{-1}, \text{ then } u', v' \in \widehat{S}$$

and the cube condition holds on $\widehat{S}$, then (S,R) is complete for reversing.

- Example: $S := \{\mathbf{a}, \mathbf{b}\}$, $R := \{\mathbf{aba} = \mathbf{bb}\}$.
  For Criterion 1: $\lambda(\mathbf{a}) := 1$, $\lambda(\mathbf{b}) := 2$;
  For Criterion 2: $\widehat{S} := \{\varepsilon, \mathbf{a}, \mathbf{b}, \mathbf{ab}, \mathbf{bb}, \mathbf{ba}, \mathbf{bba}\}$; $\rightsquigarrow$ (in both cases) OK.

- Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^+$ and of the group $\langle S, R\rangle$ can be read from R easily.

- Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^+$ and of the group $\langle S, R\rangle$ can be read from R easily.

- Proposition: Assume that (S, R) is complete for reversing, and R contains no relation $su = sv$ with $u \neq v$. Then $\langle S, R\rangle^+$ is left cancellative.

- Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^{+}$ and of the group $\langle S, R\rangle$ can be read from R easily.

- Proposition: Assume that (S, R) is complete for reversing, and R contains no relation $su = sv$ with $u \neq v$. Then $\langle S, R\rangle^{+}$ is left cancellative.

Proof: Assume $su \equiv^{+} sv$.

- Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^+$ and of the group $\langle S, R\rangle$ can be read from R easily.

- Proposition: Assume that (S, R) is complete for reversing, and R contains no relation $su = sv$ with $u \neq v$. Then $\langle S, R\rangle^+$ is left cancellative.

Proof: Assume $su \equiv^+ sv$.

Then $(su)^{-1}(sv) \curvearrowright \varepsilon$:

- Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^+$ and of the group $\langle S, R\rangle$ can be read from R easily.

- Proposition: Assume that (S, R) is complete for reversing, and R contains no relation $su = sv$ with $u \neq v$. Then $\langle S, R\rangle^+$ is left cancellative.

Proof: Assume $su \equiv^+ sv$.

Then $(su)^{-1}(sv) \curvearrowright \varepsilon$:

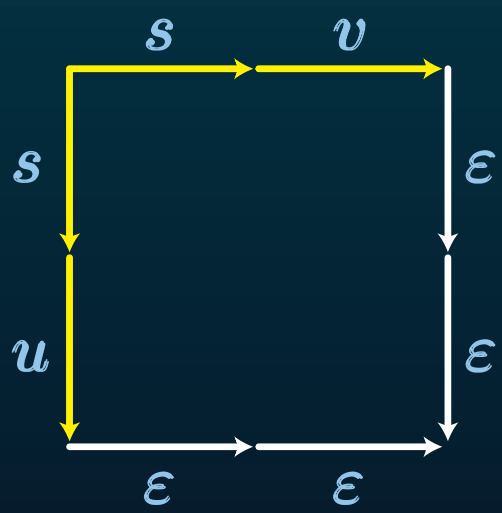
$$\begin{tikzcd} \cdot \arrow[r, "s"] \arrow[d, "s"'] & \cdot \arrow[r, "v"] & \cdot \\ \cdot \arrow[d, "u"'] & & \\ \cdot & & \end{tikzcd}$$

- Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^{+}$ and of the group $\langle S, R\rangle$ can be read from R easily.

- Proposition: Assume that (S, R) is complete for reversing, and R contains no relation $su = sv$ with $u \neq v$. Then $\langle S, R\rangle^{+}$ is left cancellative.

Proof: Assume $su \equiv^{+} sv$.

Then $(su)^{-1}(sv) \curvearrowright \varepsilon$:



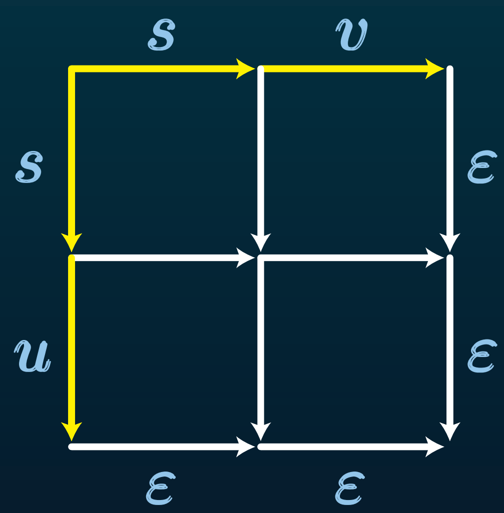
READING PROPERTIES

• Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^+$ and of the group $\langle S, R\rangle$ can be read from R easily.

• Proposition: Assume that (S, R) is complete for reversing, and R contains no relation $su = sv$ with $u \neq v$. Then $\langle S, R\rangle^+$ is left cancellative.

Proof: Assume $su \equiv^+ sv$.

Then $(su)^{-1}(sv) \curvearrowright \varepsilon$:

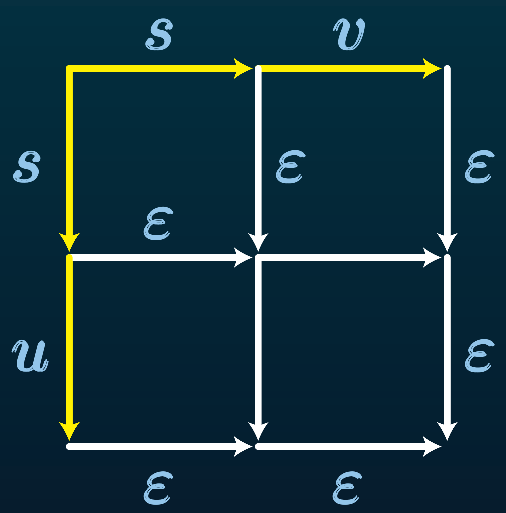


- Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^+$ and of the group $\langle S, R\rangle$ can be read from R easily.

- Proposition: Assume that (S, R) is complete for reversing, and R contains no relation $su = sv$ with $u \neq v$. Then $\langle S, R\rangle^+$ is left cancellative.

Proof: Assume $su \equiv^+ sv$.

Then $(su)^{-1}(sv) \curvearrowright \varepsilon$:

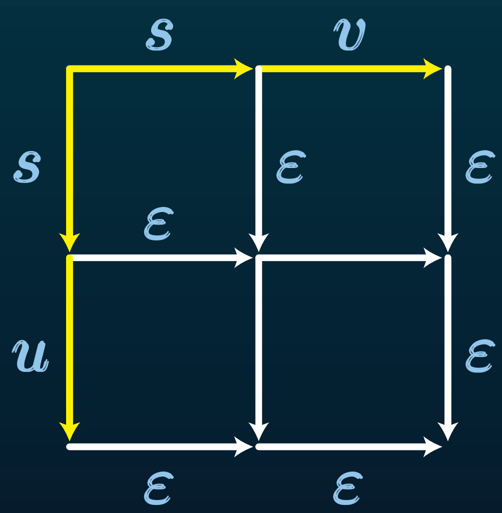


- Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^+$ and of the group $\langle S, R\rangle$ can be read from R easily.

- Proposition: Assume that (S, R) is complete for reversing, and R contains no relation $su = sv$ with $u \neq v$. Then $\langle S, R\rangle^+$ is left cancellative.

Proof: Assume $su \equiv^+ sv$.

Then $(su)^{-1}(sv) \curvearrowright \varepsilon$:



Hence $u^{-1}v \curvearrowright \varepsilon$.

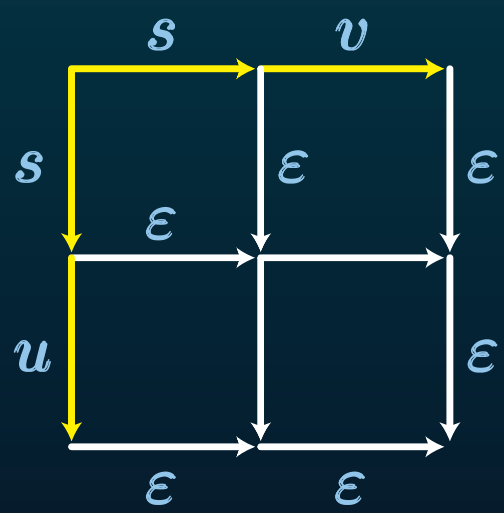
READING PROPERTIES

- Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^+$ and of the group $\langle S, R\rangle$ can be read from R easily.

- Proposition: Assume that (S, R) is complete for reversing, and R contains no relation $su = sv$ with $u \neq v$. Then $\langle S, R\rangle^+$ is left cancellative.

Proof: Assume $su \equiv^+ sv$.

Then $(su)^{-1}(sv) \curvearrowright \varepsilon$:



Hence $u^{-1}v \curvearrowright \varepsilon$.

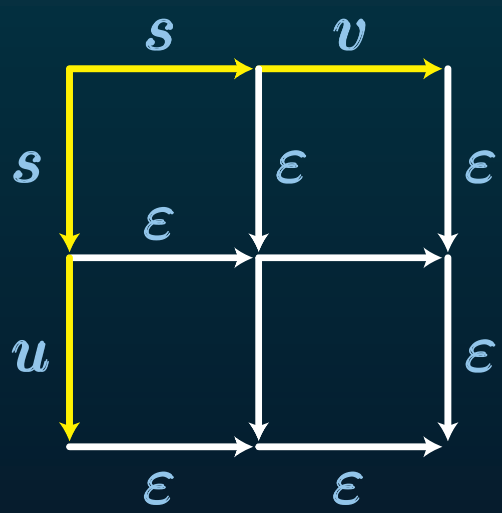
Hence $u \equiv^+ v$. $\square$

- Principle: When (S, R) is complete for reversing, the properties of the monoid $\langle S, R\rangle^+$ and of the group $\langle S, R\rangle$ can be read from R easily.

- Proposition: Assume that (S, R) is complete for reversing, and R contains no relation $su = sv$ with $u \neq v$. Then $\langle S, R\rangle^+$ is left cancellative.

Proof: Assume $su \equiv^+ sv$.

Then $(su)^{-1}(sv) \curvearrowright \varepsilon$:



Hence $u^{-1}v \curvearrowright \varepsilon$.

Hence $u \equiv^+ v$. $\square$

- Prop.: Assume that (S, R) is complete for reversing, and R contains $\leqslant 1$ relation $su = tv$ for each pair s, t in S. Then $\langle S, R\rangle^+$ admits local right lcm's.

two elements with a common multiple admit a lcm

- For $\begin{tikzcd} \bullet \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \bullet \arrow[d, "u'"] \\ \bullet \arrow[r, "v'"'] & \bullet \end{tikzcd}$, write $\mathbf{c}(u, v) := v'$ and $\boldsymbol{\delta}(u, v) := uv'$.

RECIPE

- For $\begin{tikzcd} \cdot \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \cdot \arrow[d, "u'"] \\ \cdot \arrow[r, "v'"'] & \cdot \end{tikzcd}$, write $\mathbf{c}(u, v) := v'$ and $\boldsymbol{\delta}(u, v) := uv'$.

- Algorithm: Input: A complemented presentation $(S; R)$;

- For $\begin{tikzcd} \cdot \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \cdot \arrow[d, "u'"] \\ \cdot \arrow[r, "v'"'] & \cdot \end{tikzcd}$, write $\mathbf{c}(u, v) := v'$ and $\boldsymbol{\delta}(u, v) := uv'$.

- Algorithm: Input: A complemented presentation $(S; R)$;

  1- Find the closure $\widehat{S}$ of S under $\mathbf{c}$;

- For $\begin{tikzcd} \cdot \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \cdot \arrow[d, "u'"] \\ \cdot \arrow[r, "v'"'] & \cdot \end{tikzcd}$, write $\mathbf{c}(u,v) := v'$ and $\boldsymbol{\delta}(u,v) := uv'$.

- Algorithm: Input: A complemented presentation $(S;R)$;

  1- Find the closure $\widehat{S}$ of S under $\mathbf{c}$;

  2- Check the cube condition on $\widehat{S}$;

- For
$$\begin{tikzcd} \cdot \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \cdot \arrow[d, "u'"] \\ \cdot \arrow[r, "v'"'] & \cdot \end{tikzcd}$$
, write $\mathbf{c}(u, v) := v'$ and $\boldsymbol{\delta}(u, v) := uv'$.

- Algorithm: Input: A complemented presentation $(S; R)$;
  1- Find the closure $\widehat{S}$ of S under $\mathbf{c}$;
  2- Check the cube condition on $\widehat{S}$;
  3- Find the closure $\widetilde{S}$ of $\widehat{S}$ under δ, and the maximal element w_0 of $\widetilde{S}$;

- For $$\begin{tikzcd} {} \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[d, "u'"] \\ {} \arrow[r, "v'"'] & {} \end{tikzcd}$$, write $\mathbf{c}(u, v) := v'$ and $\boldsymbol{\delta}(u, v) := uv'$.

- Algorithm: Input: A complemented presentation $(S; R)$;
  1- Find the closure $\widehat{S}$ of S under $\mathbf{c}$;
  2- Check the cube condition on $\widehat{S}$;
  3- Find the closure $\widetilde{S}$ of $\widehat{S}$ under δ, and the maximal element w_0 of $\widetilde{S}$;
  4- Check the injectivity of $u \mapsto \mathbf{c}(u, w_0)$ on $\widetilde{S}$ up to equivalence.

RECIPE

- For $\begin{tikzcd} \bullet \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \bullet \arrow[d, "u'"] \\ \bullet \arrow[r, "v'"'] & \bullet \end{tikzcd}$, write $\mathbf{c}(u, v) := v'$ and $\boldsymbol{\delta}(u, v) := uv'$.

- Algorithm: Input: A complemented presentation $(S; R)$;
  1- Find the closure $\widehat{S}$ of S under $\mathbf{c}$;
  2- Check the cube condition on $\widehat{S}$;
  3- Find the closure $\widetilde{S}$ of $\widehat{S}$ under δ, and the maximal element w_0 of $\widetilde{S}$;
  4- Check the injectivity of $u \mapsto \mathbf{c}(u, w_0)$ on $\widetilde{S}$ up to equivalence.

Then $\langle S; R \rangle^+$ is a Garside monoid with w_0 representing a Garside element.

- For $\begin{tikzcd} \cdot \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \cdot \arrow[d, "u'"] \\ \cdot \arrow[r, "v'"'] & \cdot \end{tikzcd}$, write $\mathbf{c}(u, v) := v'$ and $\boldsymbol{\delta}(u, v) := uv'$.

- Algorithm: Input: A complemented presentation $(S; R)$;
  1- Find the closure $\widehat{S}$ of S under $\mathbf{c}$;
  2- Check the cube condition on $\widehat{S}$;
  3- Find the closure $\widetilde{S}$ of $\widehat{S}$ under δ, and the maximal element w_0 of $\widetilde{S}$;
  4- Check the injectivity of $u \mapsto \mathbf{c}(u, w_0)$ on $\widetilde{S}$ up to equivalence.

Then $\langle S; R \rangle^+$ is a Garside monoid with w_0 representing a Garside element.

- Example: $S = \{\mathtt{a}, \mathtt{b}\}$, $R = \{\mathtt{aba} = \mathtt{bb}\}$.

RECIPE

- For $\begin{tikzcd} \cdot \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \cdot \arrow[d, "u'"] \\ \cdot \arrow[r, "v'"'] & \cdot \end{tikzcd}$, write $\mathbf{c}(u, v) := v'$ and $\boldsymbol{\delta}(u, v) := uv'$.

- Algorithm: Input: A complemented presentation $(S; R)$;

  1- Find the closure $\widehat{S}$ of S under $\mathbf{c}$;

  2- Check the cube condition on $\widehat{S}$;

  3- Find the closure $\widetilde{S}$ of $\widehat{S}$ under δ, and the maximal element w_0 of $\widetilde{S}$;

  4- Check the injectivity of $u \mapsto \mathbf{c}(u, w_0)$ on $\widetilde{S}$ up to equivalence.

Then $\langle S; R\rangle^+$ is a Garside monoid with w_0 representing a Garside element.

- Example: $S = \{\mathbf{a}, \mathbf{b}\}$, $R = \{\mathbf{aba} = \mathbf{bb}\}$.

  1- $\widehat{S} = \{\varepsilon, \mathbf{a}, \mathbf{b}, \mathbf{ab}, \mathbf{ba}, \mathbf{bab}\}$;   2- ... OK;

- For $\begin{tikzcd} \cdot \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \cdot \arrow[d, "u'"] \\ \cdot \arrow[r, "v'"'] & \cdot \end{tikzcd}$, write $\mathbf{c}(u, v) := v'$ and $\boldsymbol{\delta}(u, v) := uv'$.

- Algorithm: Input: A complemented presentation $(S; R)$;
  1- Find the closure $\widehat{S}$ of S under $\mathbf{c}$;
  2- Check the cube condition on $\widehat{S}$;
  3- Find the closure $\widetilde{S}$ of $\widehat{S}$ under δ, and the maximal element w_0 of $\widetilde{S}$;
  4- Check the injectivity of $u \mapsto \mathbf{c}(u, w_0)$ on $\widetilde{S}$ up to equivalence.

Then $\langle S; R\rangle^+$ is a Garside monoid with w_0 representing a Garside element.

- Example: $S = \{\mathtt{a}, \mathtt{b}\}$, $R = \{\mathtt{aba} = \mathtt{bb}\}$.
  1- $\widehat{S} = \{\varepsilon, \mathtt{a}, \mathtt{b}, \mathtt{ab}, \mathtt{ba}, \mathtt{bab}\}$; 2- ... OK;
  3- $\widetilde{S} = \widehat{S} \cup \{\mathtt{bb}, \mathtt{bbb}\}$, $w_0 = \mathtt{bbb}$; 4- $\mathbf{c}(\varepsilon, w_0) = w_0$, $\mathbf{c}(\mathtt{a}, w_0) = \mathtt{bab}$... OK

- For $\begin{tikzcd} \cdot \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \cdot \arrow[d, "u'"] \\ \cdot \arrow[r, "v'"'] & \cdot \end{tikzcd}$, write $\mathbf{c}(u, v) := v'$ and $\boldsymbol{\delta}(u, v) := uv'$.

- Algorithm: Input: A complemented presentation $(S; R)$;
  1- Find the closure $\widehat{S}$ of S under $\mathbf{c}$;
  2- Check the cube condition on $\widehat{S}$;
  3- Find the closure $\widetilde{S}$ of $\widehat{S}$ under δ, and the maximal element w_0 of $\widetilde{S}$;
  4- Check the injectivity of $u \mapsto \mathbf{c}(u, w_0)$ on $\widetilde{S}$ up to equivalence.

Then $\langle S; R\rangle^+$ is a Garside monoid with w_0 representing a Garside element.

- Example: $S = \{\mathbf{a}, \mathbf{b}\}$, $R = \{\mathbf{aba} = \mathbf{bb}\}$.
  1- $\widehat{S} = \{\varepsilon, \mathbf{a}, \mathbf{b}, \mathbf{ab}, \mathbf{ba}, \mathbf{bab}\}$; 2- ... OK;
  3- $\widetilde{S} = \widehat{S} \cup \{\mathbf{bb}, \mathbf{bbb}\}$, $w_0 = \mathbf{bbb}$; 4- $\mathbf{c}(\varepsilon, w_0) = w_0$, $\mathbf{c}(\mathbf{a}, w_0) = \mathbf{bab}$... OK

$\rightsquigarrow$ $\langle \mathbf{a}, \mathbf{b}; \mathbf{aba} = \mathbf{bb}\rangle^+$ is a Garside monoid with $\Delta = \mathbf{b}^3$ and 8 divisors of Δ.

• Proposition: Let G be a Garside group associated with (M, Δ). Then each element of G is uniquely expressed as gh^{-1} with $g, h \in M$ and $\mathrm{gcd}_R(g, h) = 1$.

- Proposition: Let G be a Garside group associated with (M, Δ). Then each element of G is uniquely expressed as gh^{-1} with $g, h \in M$ and $\mathrm{gcd}_R(g, h) = 1$.

- Algorithm: Input: A word w;

• Proposition: Let G be a Garside group associated with (M, Δ). Then each element of G is uniquely expressed as gh^{-1} with $g, h \in M$ and $\gcd_R(g, h) = 1$.

• Algorithm: Input: A word w;

1- Left reverse w into $u^{-1}v$,

$\uparrow$

symmetric to (right) reversing: $+- \to -+$

- Proposition: Let G be a Garside group associated with (M, Δ). Then each element of G is uniquely expressed as gh^{-1} with $g, h \in M$ and $\mathrm{gcd}_R(g, h) = 1$.

- Algorithm: Input: A word w;

  1- Left reverse w into $u^{-1}v$,

  $\uparrow$

  symmetric to (right) reversing: $+- \to -+$

  2- (right) reverse $u^{-1}v$ into $v'u'^{-1}$.

- Proposition: Let G be a Garside group associated with (M, Δ). Then each element of G is uniquely expressed as gh^{-1} with $g, h \in M$ and $\mathrm{gcd}_R(g, h) = 1$.

- Algorithm: Input: A word w;

  1- Left reverse w into $u^{-1}v$,

symmetric to (right) reversing: $+- \to -+$

  2- (right) reverse $u^{-1}v$ into $v'u'^{-1}$.

Output: Two positive words u', v'

s.t. $\overline{u}'\overline{v}'^{-1}$ is the above expression of $\overline{w}$.

the element of the group represented by...

COMPUTING (1): REDUCING TO THE MONOID

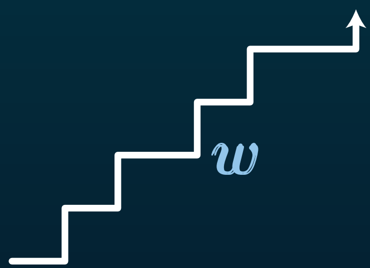
• Proposition: Let G be a Garside group associated with (M, Δ). Then each element of G is uniquely expressed as gh^{-1} with $g, h \in M$ and $\mathrm{gcd}_R(g, h) = 1$.

• Algorithm: Input: A word w;

1- Left reverse w into $u^{-1}v$,

symmetric to (right) reversing: $+- \to -+$

2- (right) reverse $u^{-1}v$ into $v'u'^{-1}$.



Output: Two positive words u', v'

s.t. $\overline{u}'\overline{v}'^{-1}$ is the above expression of $\overline{w}$.

the element of the group represented by...

COMPUTING (1): REDUCING TO THE MONOID

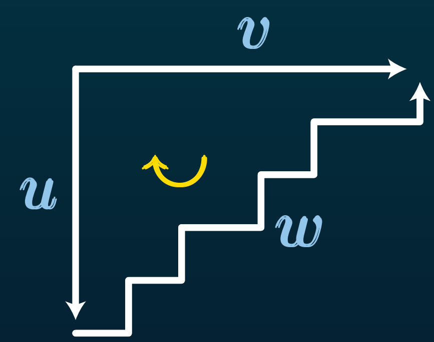
• Proposition: Let G be a Garside group associated with (M, Δ). Then each element of G is uniquely expressed as gh^{-1} with $g, h \in M$ and $\gcd_R(g, h) = 1$.

• Algorithm: Input: A word w;

1- Left reverse w into $u^{-1}v$,

symmetric to (right) reversing: $+- \rightarrow -+$

2- (right) reverse $u^{-1}v$ into $v'u'^{-1}$.



Output: Two positive words u', v'

s.t. $\overline{u}'\overline{v}'^{-1}$ is the above expression of $\overline{w}$.

the element of the group represented by...

COMPUTING (1): REDUCING TO THE MONOID

• Proposition: Let G be a Garside group associated with (M, Δ). Then each element of G is uniquely expressed as gh^{-1} with $g, h \in M$ and $\mathrm{gcd}_R(g, h) = 1$.

• Algorithm: Input: A word w;

1- Left reverse w into $u^{-1}v$,

symmetric to (right) reversing: $+- \to -+$

2- (right) reverse $u^{-1}v$ into $v'u'^{-1}$.

$$
\begin{tikzcd}
{} \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[d, "u'"] \\
{} \arrow[r, "v'"'] & {}
\end{tikzcd}
$$

Output: Two positive words u', v'

s.t. $\overline{u}'\overline{v}'^{-1}$ is the above expression of $\overline{w}$.

the element of the group represented by...

COMPUTING (1): REDUCING TO THE MONOID

• Proposition: Let G be a Garside group associated with (M, Δ). Then each element of G is uniquely expressed as gh^{-1} with $g, h \in M$ and $\gcd_R(g, h) = 1$.

• Algorithm: Input: A word w;

1- Left reverse w into $u^{-1}v$,

symmetric to (right) reversing: $+- \to -+$

2- (right) reverse $u^{-1}v$ into $v'u'^{-1}$.

$$\begin{tikzcd} \cdot \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \cdot \arrow[d, "u'"] \\ \cdot \arrow[r, "v'"'] & \cdot \end{tikzcd}$$

Output: Two positive words u', v'

s.t. $\overline{u}'\overline{v}'^{-1}$ is the above expression of $\overline{w}$.

the element of the group represented by...

• Corollary (solution to the word problem): w represents 1 iff $u' = v' = \varepsilon$.

• Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

- Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

- Algorithm: Input: Two positive words u, v;

• Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

• Algorithm: Input: Two positive words u, v;

1- Right reverse $u^{-1}v$ into $v'u'^{-1}$,

- Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

- Algorithm: Input: Two positive words u, v;

  1- Right reverse $u^{-1}v$ into $v'u'^{-1}$,
  2- Left reverse $v'u'^{-1}$ into $u''v''^{-1}$,

• Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

• Algorithm: Input: Two positive words u, v;

1- Right reverse $u^{-1}v$ into $v'u'^{-1}$,
2- Left reverse $v'u'^{-1}$ into $u''v''^{-1}$,
3- Left reverse uu''^{-1} into w.

• Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

• Algorithm: Input: Two positive words u, v;

1- Right reverse $u^{-1}v$ into $v'u'^{-1}$,
2- Left reverse $v'u'^{-1}$ into $u''v''^{-1}$,
3- Left reverse uu''^{-1} into w.

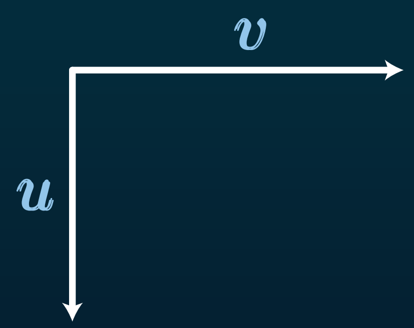
Output: uv' $(= \delta(u, v))$ and w,
representing $\mathrm{lcm}_R(\overline{u}, \overline{v})$ and $\mathrm{gcd}_L(\overline{u}, \overline{v})$ respectively.

COMPUTING (2): LATTICE OPERATIONS

• Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

• Algorithm: Input: Two positive words u, v;

1- Right reverse $u^{-1}v$ into $v'u'^{-1}$,
2- Left reverse $v'u'^{-1}$ into $u''v''^{-1}$,
3- Left reverse uu''^{-1} into w.



Output: uv' $(= \delta(u, v))$ and w,
representing $\mathrm{lcm}_R(\overline{u}, \overline{v})$ and $\mathrm{gcd}_L(\overline{u}, \overline{v})$ respectively.

COMPUTING (2): LATTICE OPERATIONS

• Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

• Algorithm: Input: Two positive words u, v;

1- Right reverse $u^{-1}v$ into $v'u'^{-1}$,
2- Left reverse $v'u'^{-1}$ into $u''v''^{-1}$,
3- Left reverse uu''^{-1} into w.

$$\begin{tikzcd} \cdot \arrow[r, "v"] \arrow[d, "u"'] \arrow[dr, phantom, "\curvearrowright"] & \cdot \arrow[d, "u'"] \\ \cdot \arrow[r, "v'"'] & \cdot \end{tikzcd}$$

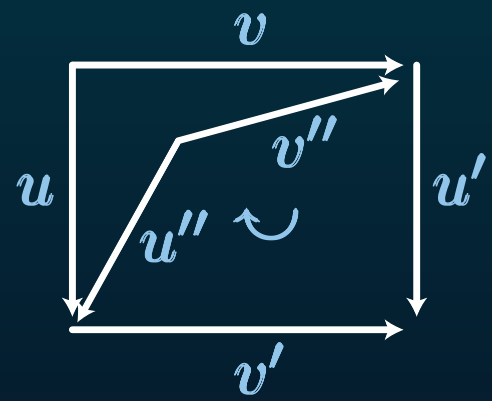
Output: uv' $(= \delta(u, v))$ and w,
representing $\mathrm{lcm}_R(\overline{u}, \overline{v})$ and $\mathrm{gcd}_L(\overline{u}, \overline{v})$ respectively.

COMPUTING (2): LATTICE OPERATIONS

• Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

• Algorithm: Input: Two positive words u, v;

1- Right reverse $u^{-1}v$ into $v'u'^{-1}$,
2- Left reverse $v'u'^{-1}$ into $u''v''^{-1}$,
3- Left reverse uu''^{-1} into w.

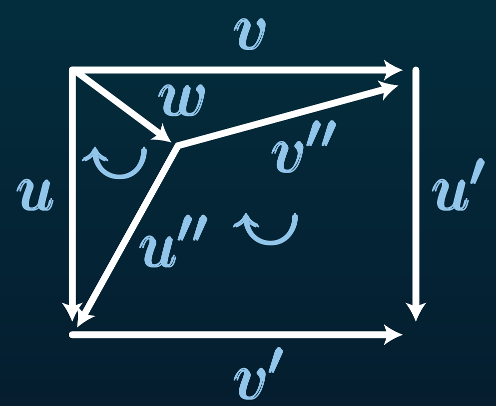


Output: uv' $(= \delta(u, v))$ and w,
representing $\mathrm{lcm}_R(\overline{u}, \overline{v})$ and $\mathrm{gcd}_L(\overline{u}, \overline{v})$ respectively.

• Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

• Algorithm: Input: Two positive words u, v;

1- Right reverse $u^{-1}v$ into $v'u'^{-1}$,
2- Left reverse $v'u'^{-1}$ into $u''v''^{-1}$,
3- Left reverse uu''^{-1} into w.

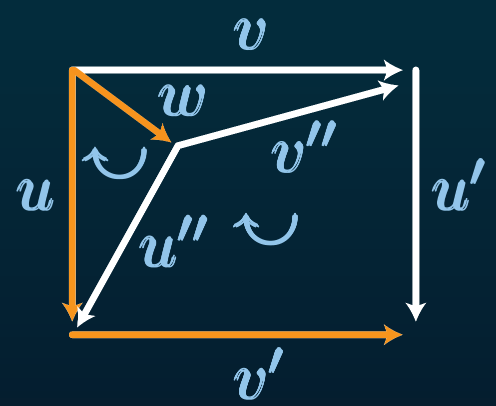


Output: uv' $(= \delta(u, v))$ and w,
representing $\mathrm{lcm}_R(\overline{u}, \overline{v})$ and $\mathrm{gcd}_L(\overline{u}, \overline{v})$ respectively.

• Proposition: Let (M, Δ) be a Garside system. Then any two elements admit a right (and a left) lcm and a left (and a right) gcd.

• Algorithm: Input: Two positive words u, v;

1- Right reverse $u^{-1}v$ into $v'u'^{-1}$,
2- Left reverse $v'u'^{-1}$ into $u''v''^{-1}$,
3- Left reverse uu''^{-1} into w.



Output: uv' $(= \delta(u, v))$ and w,
representing $\mathrm{lcm}_R(\overline{u}, \overline{v})$ and $\mathrm{gcd}_L(\overline{u}, \overline{v})$ respectively.

• Proposition: Let (M, Δ) be a Garside system. Then every element of M admits a unique expression $x_1 \ldots x_p$ with x_1, ..., x_p simple and, for each i, every simple right divisor of $x_{i-1}x_i$ is a right divisor of x_i.

↑

divisor of Δ

• Proposition: Let (M, Δ) be a Garside system. Then every element of M admits a unique expression $x_1 ... x_p$ with x_1, ..., x_p simple and, for each i, every simple right divisor of $x_{i-1}x_i$ is a right divisor of x_i.

$\uparrow$

divisor of Δ

• "Reversing is compatible with the normal form":

• Proposition: Let (M, Δ) be a Garside system. Then every element of M admits a unique expression $x_1 ... x_p$ with x_1, ..., x_p simple and, for each i, every simple right divisor of $x_{i-1}x_i$ is a right divisor of x_i.

↑ divisor of Δ

• "Reversing is compatible with the normal form":

• Proposition: Let (M, Δ) be a Garside system, and $(x_1, ..., x_p)$, $(y_1, ..., y_q)$ be normal. Then so is every horizontal– or vertical–diagonal sequence in

$$\begin{tikzcd}
{} \arrow[r, "y_1"] \arrow[d, "x_1"'] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[r, "y_2"] \arrow[d] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[r, "\cdots"] \arrow[d] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[r, "y_q"] \arrow[d] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[d] \\
{} \arrow[r] \arrow[d, "\vdots"'] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[r] \arrow[d] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[r] \arrow[d] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[r] \arrow[d] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[d] \\
{} \arrow[r] \arrow[d, "x_p"'] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[r] \arrow[d] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[r] \arrow[d] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[r] \arrow[d] \arrow[dr, phantom, "\curvearrowright"] & {} \arrow[d] \\
{} \arrow[r] & {} \arrow[r] & {} \arrow[r] & {} \arrow[r] & {}
\end{tikzcd} .$$

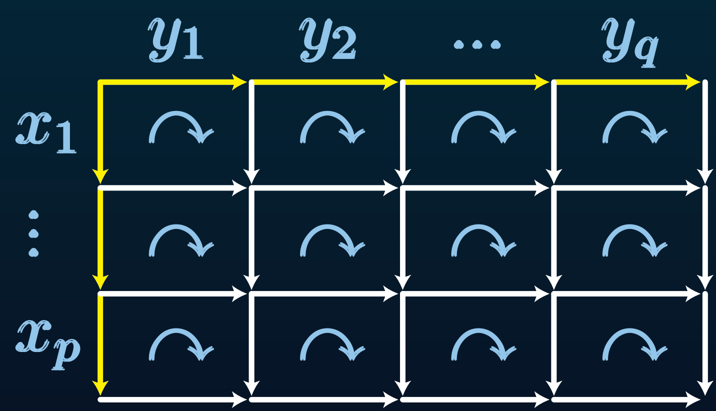
COMPUTING (3): NORMAL FORM

- Proposition: Let (M, Δ) be a Garside system. Then every element of M admits a unique expression $x_1 ... x_p$ with x_1, ..., x_p simple and, for each i, every simple right divisor of $x_{i-1}x_i$ is a right divisor of x_i.

  ↑ divisor of Δ

- "Reversing is compatible with the normal form":

- Proposition: Let (M, Δ) be a Garside system, and $(x_1, ..., x_p)$, $(y_1, ..., y_q)$ be normal. Then so is every horizontal– or vertical–diagonal sequence in



.

• Proposition: Let (M, Δ) be a Garside system. Then every element of M admits a unique expression $x_1 ... x_p$ with x_1, ..., x_p simple and, for each i, every simple right divisor of $x_{i-1}x_i$ is a right divisor of x_i.

↑ divisor of Δ

• "Reversing is compatible with the normal form":

• Proposition: Let (M, Δ) be a Garside system, and $(x_1, ..., x_p)$, $(y_1, ..., y_q)$ be normal. Then so is every horizontal– or vertical–diagonal sequence in

.

• Proposition: Let (M, Δ) be a Garside system. Then every element of M admits a unique expression $x_1 ... x_p$ with x_1, ..., x_p simple and, for each i, every simple right divisor of $x_{i-1}x_i$ is a right divisor of x_i.

$\uparrow$ divisor of Δ

• "Reversing is compatible with the normal form":

• Proposition: Let (M, Δ) be a Garside system, and $(x_1, ..., x_p)$, $(y_1, ..., y_q)$ be normal. Then so is every horizontal– or vertical–diagonal sequence in

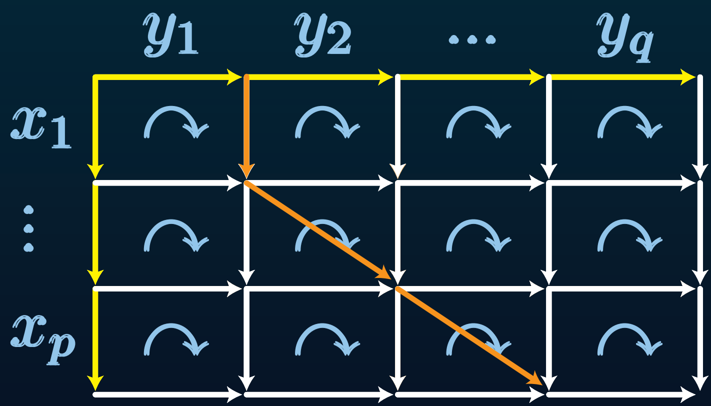
.

• Proposition: Let (M, Δ) be a Garside system. Then every element of M admits a unique expression $x_1 ... x_p$ with x_1, ..., x_p simple and, for each i, every simple right divisor of $x_{i-1} x_i$ is a right divisor of x_i.

↑ divisor of Δ

• "Reversing is compatible with the normal form":

• Proposition: Let (M, Δ) be a Garside system, and $(x_1, ..., x_p)$, $(y_1, ..., y_q)$ be normal. Then so is every horizontal– or vertical–diagonal sequence in



.

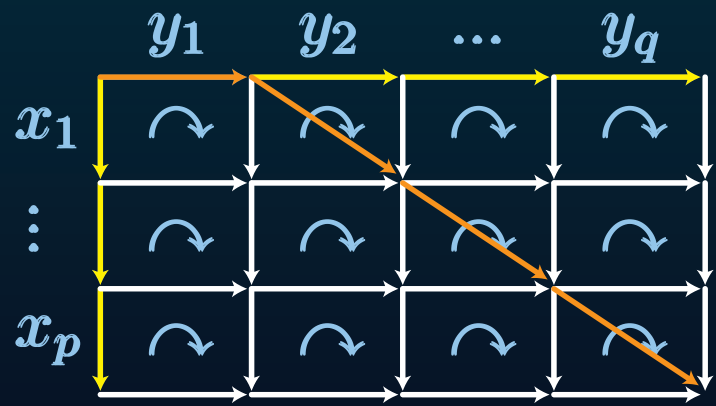
COMPUTING (3): NORMAL FORM

• Proposition: Let (M, Δ) be a Garside system. Then every element of M admits a unique expression $x_1 ... x_p$ with x_1, ..., x_p simple and, for each i, every simple right divisor of $x_{i-1}x_i$ is a right divisor of x_i.

↑ divisor of Δ

• "Reversing is compatible with the normal form":

• Proposition: Let (M, Δ) be a Garside system, and $(x_1, ..., x_p)$, $(y_1, ..., y_q)$ be normal. Then so is every horizontal– or vertical–diagonal sequence in



.

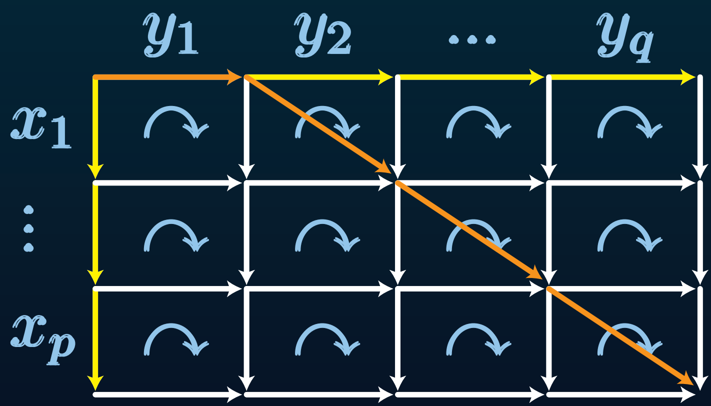
COMPUTING (3): NORMAL FORM

• Proposition: Let (M, Δ) be a Garside system. Then every element of M admits a unique expression $x_1 \ldots x_p$ with x_1, ..., x_p simple and, for each i, every simple right divisor of $x_{i-1} x_i$ is a right divisor of x_i.

↑ divisor of Δ

• "Reversing is compatible with the normal form":

• Proposition: Let (M, Δ) be a Garside system, and $(x_1, ..., x_p)$, $(y_1, ..., y_q)$ be normal. Then so is every horizontal– or vertical–diagonal sequence in



.

$\rightsquigarrow$ Computation of the normal form of a product, or of an lcm.

- Word reversing is a convenient tool

  - for recognizing a Garside group from a (complemented) presentation,

  - for computing inside a Garside group using a (complemented) presentation for (one of the possible) Garside structure(s).

- Word reversing is a convenient tool
  - for recognizing a Garside group from a (complemented) presentation,
  - for computing inside a Garside group using a (complemented) presentation for (one of the possible) Garside structure(s).

- Once completeness is granted, reversing $\approx$ computing lcm, but not a priori
  $\rightsquigarrow$ crucial to distinguish between words and the elements they represent.

- Word reversing is a convenient tool
  - for recognizing a Garside group from a (complemented) presentation,
  - for computing inside a Garside group using a (complemented) presentation for (one of the possible) Garside structure(s).

- Once completeness is granted, reversing $\approx$ computing lcm, but not a priori
  $\rightsquigarrow$ crucial to distinguish between words and the elements they represent.

- References:

- Groupes de Garside; Ann. Scient. Ec. Norm. Sup. 35 (2002) 267–306.
- Complete positive group presentations; J. of Algebra 268 (2003) 156–197.

- Word reversing is a convenient tool

  - for recognizing a Garside group from a (complemented) presentation,

  - for computing inside a Garside group using a (complemented) presentation for (one of the possible) Garside structure(s).

- Once completeness is granted, reversing $\approx$ computing lcm, but not a priori

  $\rightsquigarrow$ crucial to distinguish between words and the elements they represent.

- References:

- Groupes de Garside; Ann. Scient. Ec. Norm. Sup. 35 (2002) 267–306.
- Complete positive group presentations; J. of Algebra 268 (2003) 156–197.

www.math.unicaen.fr/$\sim$dehornoy