Completion of operadic rewriting systems by Gaussian elimination

Benjamin Dupont - Philippe Malbos - Isaac Ren

International Workshop on Confluence 2021

July 23rd, 2021

Outline

I. Introduction and motivation

II. Rewriting in non-symmetric operads

III. F4 completion procedure for non-symmetric operads

IV. Case study: the anti-associative operad

I. Introduction and motivation

Elimination theory

- **Algebraic rewriting :** study of presentations of algebraic structures by generators and oriented relations.
  - Solve decision problems, *e.g.* the **word problem**.
  - Symbolic computation methods: homological invariants, resolutions, cofibrant replacements.
  - Methods to solve systems of polynomial equations.

Elimination theory

- **Algebraic rewriting :** study of presentations of algebraic structures by generators and oriented relations.
  - Solve decision problems, *e.g.* the **word problem**.
  - Symbolic computation methods: homological invariants, resolutions, cofibrant replacements.
  - Methods to solve systems of polynomial equations.
- This latter has been developed in **elimination theory**.
  - **Gaussian elimination** is used to solve linear systems by eliminating indeterminates.
  - **Gröbner bases** give methods to solve non-linear systems by elimination.

Elimination theory

- **Algebraic rewriting :** study of presentations of algebraic structures by generators and oriented relations.
  - Solve decision problems, *e.g.* the **word problem**.
  - Symbolic computation methods: homological invariants, resolutions, cofibrant replacements.
  - Methods to solve systems of polynomial equations.
- This latter has been developed in **elimination theory**.
  - **Gaussian elimination** is used to solve linear systems by eliminating indeterminates.
  - **Gröbner bases** give methods to solve non-linear systems by elimination.
- **Objective:** Compute Gröbner bases by completion of a set of polynomials.
  - Gröbner bases are confluent linear rewriting systems compatible with a monomial order.
  - Computing a Gröbner basis is equivalent to completing a linear rewriting system.

Convergence and Gröbner bases

- Let X be an alphabet, and X^* be the free monoid on X.
- Fix a **monomial order** $\prec$ on X^*, that is a well founded total order such that $u \prec v$ implies $wuw' \prec wvw'$ for any w, w' in X^*.

Convergence and Gröbner bases

- Let X be an alphabet, and X^* be the free monoid on X.
- Fix a **monomial order** $\prec$ on X^*, that is a well founded total order such that $u \prec v$ implies $wuw' \prec wvw'$ for any w, w' in X^*.
- Any polynomial u on the variables of X admits a unique **leading monomial**, denoted by $lm(u)$. It admits a leading coefficient $lc(u)$.

Convergence and Gröbner bases

- Let X be an alphabet, and X^* be the free monoid on X.

- Fix a **monomial order** $\prec$ on X^*, that is a well founded total order such that $u \prec v$ implies $wuw' \prec wvw'$ for any w, w' in X^*.

- Any polynomial u on the variables of X admits a unique **leading monomial**, denoted by $lm(u)$. It admits a leading coefficient $lc(u)$.

- A **Gröbner basis** for an ideal I generated by a set R of relations compatible with $\prec$ is a subset G of I such that
$$\langle lm(I) \rangle = \langle lm(G) \rangle$$

Convergence and Gröbner bases

- Let X be an alphabet, and X^* be the free monoid on X.
- Fix a **monomial order** $\prec$ on X^*, that is a well founded total order such that $u \prec v$ implies $wuw' \prec wvw'$ for any w, w' in X^*.
- Any polynomial u on the variables of X admits a unique **leading monomial**, denoted by $lm(u)$. It admits a leading coefficient $lc(u)$.
- A **Gröbner basis** for an ideal I generated by a set R of relations compatible with $\prec$ is a subset G of I such that
$$\langle lm(I) \rangle = \langle lm(G) \rangle$$
- **Theorem :** A subset G of I is a Gröbner basis if and only if the rewriting system $\longrightarrow_G$ whose rules are
$$lm(u) \longrightarrow_G lm(u) - \frac{1}{lc(u)} u$$
for each u in G is convergent.

Linear rewriting

- Linear rewriting in algebra:
  - for commutative algebras, **Janet '20**, **Buchberger '65**,
  - for associative algebras, **Bokut '76**, **Bergman '78**, **Mora '94**,
  - for operads, **Dotsenko-Khoroshkin '2010**, **Malbos-Ren '2021**,

Linear rewriting

- Linear rewriting in algebra:
  - for commutative algebras, **Janet '20**, **Buchberger '65**,
  - for associative algebras, **Bokut '76**, **Bergman '78**, **Mora '94**,
  - for operads, **Dotsenko-Khoroshkin '2010**, **Malbos-Ren '2021**,
- Improved completion procedures for Gröbner bases of commutative algebras, *e.g.*
  - **Buchberger**'s **syzygy criterion** in 1979,
  - **Faugère** **F4 and F5 algorithms** in 1999 and 2002.

Linear rewriting

- Linear rewriting in algebra:
  - for commutative algebras, **Janet '20**, **Buchberger '65**,
  - for associative algebras, **Bokut '76**, **Bergman '78**, **Mora '94**,
  - for operads, **Dotsenko-Khoroshkin '2010**, **Malbos-Ren '2021**,
- Improved completion procedures for Gröbner bases of commutative algebras, *e.g.*
  - **Buchberger**'s **syzygy criterion** in 1979,
  - **Faugère** **F4 and F5 algorithms** in 1999 and 2002.
- These procedures are based on two principles:
  - eliminate non-necessary branchings, *e.g.* **orthogonal branchings**,
  - eliminate redundant critical branchings using **syzygies**.

Linear rewriting

- Linear rewriting in algebra:
  - for commutative algebras, **Janet '20**, **Buchberger '65**,
  - for associative algebras, **Bokut '76**, **Bergman '78**, **Mora '94**,
  - for operads, **Dotsenko-Khoroshkin '2010**, **Malbos-Ren '2021**,
- Improved completion procedures for Gröbner bases of commutative algebras, *e.g.*
  - **Buchberger**'s **syzygy criterion** in 1979,
  - **Faugère** **F4 and F5 algorithms** in 1999 and 2002.
- These procedures are based on two principles:
  - eliminate non-necessary branchings, *e.g.* **orthogonal branchings**,
  - eliminate redundant critical branchings using **syzygies**.
- These approaches have been studied in the case of non-commutative algebras, **Xiu '12**, **Chenavier '19**, **Hofstadler '20**.
- **Objective:** Extend these constructions for the case of non-symmetric operads.

II. Rewriting in non-symmetric operads

Non-symmetric operads

- A (*non-symmetric*) **operad** is a collection of vector spaces $(P(k))_{k\geq 1}$ graded by **arity** and equipped with *partial composition maps*

$$\circ_i : P(m) \circ_i P(n) \to P(m+n-1)$$

for all $m, n \geq 1$ and $1 \leq i \leq m$.

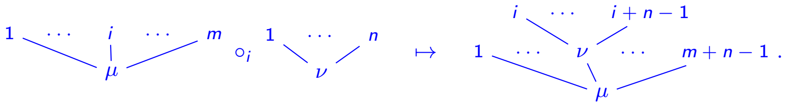
Non-symmetric operads

- A (*non-symmetric*) **operad** is a collection of vector spaces $(P(k))_{k\geq 1}$ graded by **arity** and equipped with *partial composition maps*

$$\circ_i : P(m) \circ_i P(n) \to P(m+n-1)$$

for all $m, n \geq 1$ and $1 \leq i \leq m$.

- Graphically, this can be represented by planar trees, called **tree monomials**:



The set of tree monomials is denoted by $\mathcal{T}(\Sigma)$.

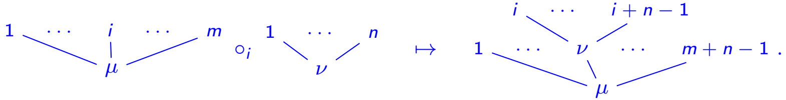
Non-symmetric operads

- A (*non-symmetric*) **operad** is a collection of vector spaces $(P(k))_{k\geq 1}$ graded by **arity** and equipped with *partial composition maps*

$$\circ_i : P(m) \circ_i P(n) \to P(m+n-1)$$

  for all $m, n \geq 1$ and $1 \leq i \leq m$.

- Graphically, this can be represented by planar trees, called **tree monomials**:

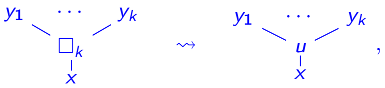


  The set of tree monomials is denoted by $\mathcal{T}(\Sigma)$.

- A **free operad** is spanned by such planar trees, whose inner vertices belong to a graded generating set $\Sigma = (\Sigma(k))_{k\geq 1}$. It is denoted by $\mathcal{F}(\Sigma)$.

Contexts

- A (**monomial**) **context of inner arity** k is a planar tree $C[\square_k]$ whose inner vertices are in Σ except for one, a symbol $\square_k$ of arity k.
- For a tree monomial u of arity k, we define the tree monomial $C[u]$ by replacing $\square_k$ with u:



with $x, y_1, \ldots, y_k$ tree monomials.

Monomial orders for operads

- An **operadic monomial order** is a total order $\prec$ on planar trees such that, for all tree monomials u, u', v, v' and appropriate compositions $\circ_i$,

$$u \prec u', v \prec v' \quad \Rightarrow \quad u \circ_i u' \prec v \circ_i v'$$

Monomial orders for operads

- An **operadic monomial order** is a total order $\prec$ on planar trees such that, for all tree monomials u, u', v, v' and appropriate compositions $\circ_i$,

$$u \prec u', v \prec v' \quad \Rightarrow \quad u \circ_i u' \prec v \circ_i v'$$

- This monomial order generalizes to rewriting rules in context: $C[\alpha]$ is a planar tree over $\Sigma \cup \{\alpha\}$. This allows us to compare rewriting rules in context.

Operadic rewriting systems (ORS)

- An **operadic rewriting system** (ORS) is a data (Σ, R) made of a graded set Σ and a binary relation $R \subset \mathcal{T}(\Sigma) \times \mathcal{F}(\Sigma)$.

- We consider ORS that are **compatible with a monomial order**, that is there is a monomial order $\prec$ such that $h \prec s(\alpha)$ for all $\alpha \in R$ and any tree monomial h in $t(\alpha)$.

Operadic rewriting systems (ORS)

- An **operadic rewriting system** (ORS) is a data (Σ, R) made of a graded set Σ and a binary relation $R \subset \mathcal{T}(\Sigma) \times \mathcal{F}(\Sigma)$.
- We consider ORS that are **compatible with a monomial order**, that is there is a monomial order $\prec$ such that $h \prec s(\alpha)$ for all $\alpha \in R$ and any tree monomial h in $t(\alpha)$.
- A reduction of the form $\lambda C[a] + 1_g : \lambda C[s(\alpha)] + g \longrightarrow_R \lambda C[t(\alpha)] + g$ is a:
  - **rewriting monomial** if $\lambda = 1$ and $g = 0$.
  - **rewriting step** if $C[s(\alpha)]$ does not appear as a monomial in g.

Operadic rewriting systems (ORS)

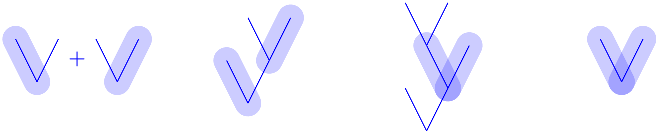
- An **operadic rewriting system** (ORS) is a data (Σ, R) made of a graded set Σ and a binary relation $R \subset \mathcal{T}(\Sigma) \times \mathcal{F}(\Sigma)$.
- We consider ORS that are **compatible with a monomial order**, that is there is a monomial order $\prec$ such that $h \prec s(\alpha)$ for all $\alpha \in R$ and any tree monomial h in $t(\alpha)$.
- A reduction of the form $\lambda C[a] + 1_g : \lambda C[s(\alpha)] + g \longrightarrow_R \lambda C[t(\alpha)] + g$ is a:
  - **rewriting monomial** if $\lambda = 1$ and $g = 0$.
  - **rewriting step** if $C[s(\alpha)]$ does not appear as a monomial in g.
- The data $(\mathcal{F}(\Sigma), R^{\mathsf{stp}})$ defines a terminating ARS.

Confluence of ORS

- A **branching** (resp. **local branching**) of (Σ, R) is a pair of rewriting paths (resp. rewriting steps) (α, β) of (Σ, R) such that $s(\alpha) = s(\beta)$.

- Local branchings :

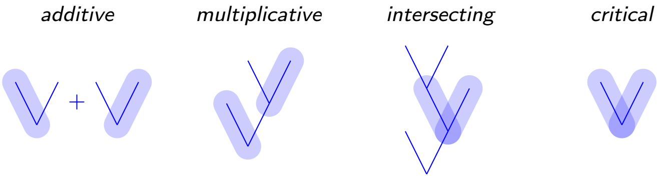
additive *multiplicative* *intersecting* *critical*



Confluence of ORS

- A **branching** (resp. **local branching**) of (Σ, R) is a pair of rewriting paths (resp. rewriting steps) (α, β) of (Σ, R) such that $s(\alpha) = s(\beta)$.

- Local branchings :



- **Critical branching lemma:** An ORS (Σ, R) compatible with $\prec$ is locally confluent if and only if its critical branchings are confluent.

- One can furthermore restrict the set of branchings to consider: **essential branchings** are "atomic" critical branchings.

- **Theorem (Malbos-R. '21):** An ORS (Σ, R) compatible with $\prec$ is locally confluent if and only if its essential branchings are confluent.

III. F4 completion procedure for non-symmetric operads

Gröbner basis completion procedures

- The general procedure is as follows:

Input: A set R of rules of a terminating rewriting system.

Set $R' := R$.

Let $\mathcal{C}$ be the set of **critical branchings** of R'.

while $\mathcal{C} \neq \varnothing$ **do**

  Select a subset B of branchings in $\mathcal{C}$, and remove them from $\mathcal{C}$.

  Add rewriting rules to R' to make the non-confluent branchings of B confluent.

  Update $\mathcal{C}$ with critical branchings induced by the new rules.

Output: A set R' of rules of a confluent rewriting system.

We study two points:

- Instead of critical branchings, we can choose any **confluence obstruction map**.
- Choosing a non-singleton subset for B implies **parallel completion** of R'.

Confluence obstruction map

Idea: if the branchings of the confluence obstruction map are confluent, then the entire rewriting system is confluent.

- A map $\mathcal{CO}$ that associates to every ORS X a set of branchings $\mathcal{CO}(X)$ of X is a **confluence obstruction map** when, for every terminating ORS X,

$$X \text{ is } \mathcal{CO}(X)\text{-confluent} \quad \Leftrightarrow \quad X \text{ is confluent.}$$

Confluence obstruction map

Idea: if the branchings of the confluence obstruction map are confluent, then the entire rewriting system is confluent.

- A map $\mathcal{CO}$ that associates to every ORS X a set of branchings $\mathcal{CO}(X)$ of X is a **confluence obstruction map** when, for every terminating ORS X,

$$X \text{ is } \mathcal{CO}(X)\text{-confluent} \quad \Leftrightarrow \quad X \text{ is confluent.}$$

- A "good" confluence obstruction map is a **confluence generating set**.
  - For example, **critical branching lemma** states that critical branchings form a confluence obstruction map.

Confluence obstruction map

Idea: if the branchings of the confluence obstruction map are confluent, then the entire rewriting system is confluent.

- A map $\mathcal{CO}$ that associates to every ORS X a set of branchings $\mathcal{CO}(X)$ of X is a **confluence obstruction map** when, for every terminating ORS X,

$$X \text{ is } \mathcal{CO}(X)\text{-confluent} \quad \Leftrightarrow \quad X \text{ is confluent.}$$

- A "good" confluence obstruction map is a **confluence generating set**.
  - For example, **critical branching lemma** states that critical branchings form a confluence obstruction map.
- A set B of branchings of an ORS X is **confluence-generating** if, for any branching (f, g) of X, there exist
  - branchings $(f_1, g_1), \ldots, (f_n, g_n)$, which are additive, multiplicative, or in B,
  - rewriting paths f' and g',
  - contexts $C_1, \ldots, C_n$,

  such that $f = C_1[f_1] \cdot f'$, $g = C_n[g_n] \cdot g'$, and for all $1 \leq i \leq n-1$, $C_i[g_i] = C_{i+1}[f_{i+1}]$.

Confluence obstruction map

Idea: if the branchings of the confluence obstruction map are confluent, then the entire rewriting system is confluent.

- A map $\mathcal{CO}$ that associates to every ORS X a set of branchings $\mathcal{CO}(X)$ of X is a **confluence obstruction map** when, for every terminating ORS X,

$$X \text{ is } \mathcal{CO}(X)\text{-confluent} \quad \Leftrightarrow \quad X \text{ is confluent.}$$

- A "good" confluence obstruction map is a **confluence generating set**.
  - For example, **critical branching lemma** states that critical branchings form a confluence obstruction map.
- A set B of branchings of an ORS X is **confluence-generating** if, for any branching (f, g) of X, there exist
  - branchings $(f_1, g_1), \ldots, (f_n, g_n)$, which are additive, multiplicative, or in B,
  - rewriting paths f' and g',
  - contexts $C_1, \ldots, C_n$,

  such that $f = C_1[f_1] \cdot f'$, $g = C_n[g_n] \cdot g'$, and for all $1 \leq i \leq n-1$, $C_i[g_i] = C_{i+1}[f_{i+1}]$.
- The essential branchings form a smaller confluence generating set: this is what we will use.

Parallel completion

In 1995, Faugère introduced a procedure called **F4** to parallelize the completion of branchings for polynomial rings.

- At each iteration: select a subset of branchings B following a **selection strategy** $\mathcal{S}$.

Parallel completion

In 1995, Faugère introduced a procedure called **F4** to parallelize the completion of branchings for polynomial rings.

- At each iteration: select a subset of branchings B following a **selection strategy** $\mathcal{S}$.
- Calculate the list of rewriting monomials that appear in the reduction of the selected branchings:

Input: An ORS (Σ, R) compatible with $\prec$, a list of branchings B, a reduction strategy σ.
Set $R' := \cup_{(f,g)\in B}\{f, g\}$.
Set $T := \cup_{f\in R'}\mathrm{supp}(t(f))$.
while $T \neq \varnothing$ **do**
  Select a monomial u in T.
  if $\sigma(u)$ is not an identity **then**
    Add the rewriting monomial $\sigma(u)$ to R'.
    Add $\mathrm{supp}(t(\sigma(u)))$ to T.
Output: A list of rewriting monomials R'.

Parallel completion

In 1995, Faugère introduced a procedure called **F4** to parallelize the completion of branchings for polynomial rings.

- At each iteration: select a subset of branchings B following a **selection strategy** $\mathcal{S}$.
- Calculate the list of rewriting monomials that appear in the reduction of the selected branchings:

Input: An ORS (Σ, R) compatible with $\prec$, a list of branchings B, a reduction strategy σ.
Set $R' := \cup_{(f,g)\in B}\{f, g\}$.
Set $T := \cup_{f\in R'}\mathrm{supp}(t(f))$.
while $T \neq \varnothing$ **do**
  Select a monomial u in T.
  if $\sigma(u)$ is not an identity **then**
    Add the rewriting monomial $\sigma(u)$ to R'.
    Add $\mathrm{supp}(t(\sigma(u)))$ to T.
Output: A list of rewriting monomials R'.

- Construct a matrix $M_{R'}$ whose rows are the rewriting monomials of R', written in the basis of tree monomials that occur.

Parallel completion

In 1995, Faugère introduced a procedure called **F4** to parallelize the completion of branchings for polynomial rings.

- At each iteration: select a subset of branchings B following a **selection strategy** $\mathcal{S}$.
- Calculate the list of rewriting monomials that appear in the reduction of the selected branchings:

Input: An ORS (Σ, R) compatible with $\prec$, a list of branchings B, a reduction strategy σ.
Set $R' := \cup_{(f,g)\in B}\{f, g\}$.
Set $T := \cup_{f\in R'}\mathrm{supp}(t(f))$.
while $T \neq \varnothing$ **do**
  Select a monomial u in T.
  if $\sigma(u)$ is not an identity **then**
    Add the rewriting monomial $\sigma(u)$ to R'.
    Add $\mathrm{supp}(t(\sigma(u)))$ to T.
Output: A list of rewriting monomials R'.

- Construct a matrix $M_{R'}$ whose rows are the rewriting monomials of R', written in the basis of tree monomials that occur.
- Reduce $M_{R'}$ to its row echelon form: the rows whose leading monomials are not sources of rewriting rules in R' are the new rules that we add to R'.

Improved completion procedure

- In summary: we have parametrized the completion procedure by a confluence obstruction map $\mathcal{CO}$, a selection strategy $\mathcal{S}$.

Input: A set R of rules of a terminating rewriting system.
Set $R' := R$.
Let $\mathcal{C} := \mathcal{CO}(R')$**.**
while $\mathcal{C} \neq \varnothing$ **do**
  Select $B := \mathcal{S}(\mathcal{C})$, and remove them from $\mathcal{C}$.
  Following F4, add rewriting rules to R' to make the non-confluent branchings of B confluent.
  Update $\mathcal{C}$ with the branchings of $\mathcal{CO}(R')$ induced by the new rules.
Output: A set R' of rules of a confluent rewriting system.

IV. Case study: the anti-associative operad

- Consider the following ORS that presents the **anti-associative operad**:

$$X := \left\langle x \in \Sigma(2) \;\middle|\; f : x(x(1,2),3) \to -\, x(1,x(2,3)) \right\rangle.$$

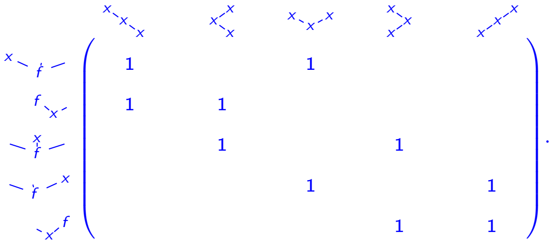
- We study the execution of our completion procedure with:
  1. the confluence obstruction map that selects **essential branchings**,
  2. the selection strategy that selects the **branchings of lowest weight**,
  3. the path-lexicographic monomial order $\prec$,
  4. the reduction strategy σ given by taking the smallest rewriting monomial for the context path-lexicographic order defined in [Malbos-R. 2021].

Case study: antiassociative operad

- First iteration: we select the only essential branching $(f \circ_1 x, x \circ_1 f)$.
- We obtain the list of rewriting monomials

$$R' = \{x \circ_1 f, f \circ_2 x, x \circ_2 f, f \circ_1 x, f \circ_3 x\}.$$

- The matrix $M_{R'}$ is of the form



$$\begin{pmatrix} 1 & & 1 & & \\ 1 & 1 & & & \\ & 1 & & 1 & \\ & & 1 & & 1 \\ & & & 1 & 1 \end{pmatrix}.$$

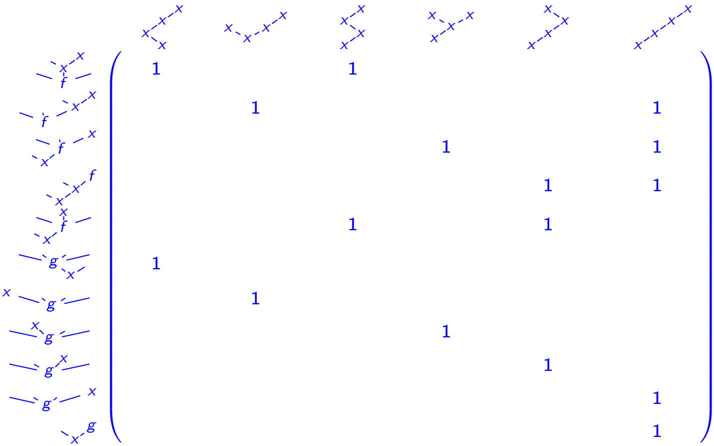
- Row reduction gives 1s on the diagonal; we add one rewriting rule $g : x \circ_2 (x \circ_2 x) \to 0$ to the ORS.

Case study: associative operad

- Second iteration: we select all five essential branchings

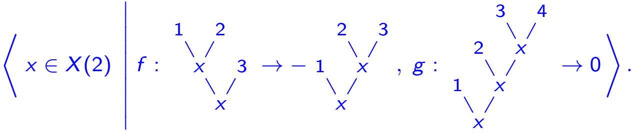
$$P := \{(f \circ_2 (x \circ_2 x), x \circ_1 g), (f \circ_3 (x \circ_2 x), g \circ_1 x), (x \circ_2 (f \circ_3 x), g \circ_2 x),$$
$$(x \circ_2 (x \circ_2 f), g \circ_3 x), (x \circ_2 g, g \circ_4 x)\}.$$

- The matrix $M_{R'}$ is



$$\begin{pmatrix}
1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

- Each column corresponds to the leading monomial of a rewriting rule in R', so there are no new rewriting rules.
- The procedure terminates and the final convergent presentation is



Outlook

We have improved the completion procedure for non-symmetric operads in two ways:

- by reducing the number of branchings to be completed,
- by parallelizing the procedure.

And now...

- Investigate better choices for the confluence obstruction map.
- Apply this completion procedure to other monoidal structures, such as **properads**.
- Interpret completion **modulo** the linear structure.

Thank you !

Essential branchings

- Let us fix an ORS (Σ, R) and a monomial order $\prec$ on $\mathcal{T}(\Sigma)$.
- A monomial order on $\mathcal{T}(\Sigma \cup \{\square_k\}_{k\geq 1})$ induces a monomial order on contexts of $\mathcal{F}(\Sigma)$.
- Given
  - a monomial order $\prec$ on $\mathcal{T}(\Sigma)$,
  - a monomial order on contexts,
  - a total order $<$ on R,

  we define the **rewriting monomial order** $\prec_{\mathbf{rm}}$ on the set of rewriting monomials by setting $C[\alpha] \prec_{\mathbf{rm}} D[\beta]$ iff
  - $C[s(\alpha)] \prec D[s(\beta)]$, or
  - $C[s(\alpha)] = D[s(\beta)]$ and $C \prec D$,
  - $C[s(\alpha)] = D[s(\beta)]$, $C = D$ and $\alpha < \beta$.
- An **essential branching** for (Σ, R) is a critical branching $(C[\alpha], D[\beta])$ s.t. $C[\alpha] \prec_{\mathbf{rm}} D[\beta]$ and they are consecutive for this order, **i.e.** there does not exist a rewriting monomial $E[\gamma]$ such that

$$C[\alpha] \prec_{\mathbf{rm}} E[\gamma] \prec_{\mathbf{rm}} D[\beta].$$