The diamond lemma for free modules

Cyrille Chenavier

Université Paris-Est Marne-la-Vallée

Laboratoire d'informatique Gaspard-Monge

IWC 2018

July 7, 2018, Oxford

Plan

Plan

I. Motivations

The diamond lemma

- The diamond lemma relates confluence and local confluence properties for terminating (abstract) rewriting systems.

The diamond lemma

- The diamond lemma relates confluence and local confluence properties for terminating (abstract) rewriting systems.
  - It provides an effective criterion to check the confluence property

The diamond lemma

- The diamond lemma relates confluence and local confluence properties for terminating (abstract) rewriting systems.
  - It provides an effective criterion to check the confluence property: the **CP Theorem**.

The diamond lemma

- The diamond lemma relates confluence and local confluence properties for terminating (abstract) rewriting systems.
  - It provides an effective criterion to check the confluence property: the CP Theorem.
- The CP theorem is formulated in various contexts

The diamond lemma

- The diamond lemma relates confluence and local confluence properties for terminating (abstract) rewriting systems.
  - It provides an effective criterion to check the confluence property: the CP Theorem.
- The CP theorem is formulated in various contexts, e.g.
  - Term rewriting, word rewriting, Gröbner bases, $\cdots$

The diamond lemma

- The diamond lemma relates confluence and local confluence properties for terminating (abstract) rewriting systems.
  - It provides an effective criterion to check the confluence property: the CP Theorem.
- The CP theorem is formulated in various contexts, e.g.
  - Term rewriting, word rewriting, Gröbner bases, $\cdots$
- If each critical pair is confluent, the normal form procedure can be used to solve decision problems

The diamond lemma

- The diamond lemma relates confluence and local confluence properties for terminating (abstract) rewriting systems.
  - It provides an effective criterion to check the confluence property: the CP Theorem.
- The CP theorem is formulated in various contexts, e.g.
  - **Term rewriting**, word rewriting, Gröbner bases, $\cdots$
- If each critical pair is confluent, the normal form procedure can be used to solve decision problems, e.g.
  - **Decide if an equation is valid for an equational theory**

The diamond lemma

- The diamond lemma relates confluence and local confluence properties for terminating (abstract) rewriting systems.
  - It provides an effective criterion to check the confluence property: the CP Theorem.
- The CP theorem is formulated in various contexts, e.g.
  - Term rewriting, **word rewriting**, Gröbner bases, $\cdots$
- If each critical pair is confluent, the normal form procedure can be used to solve decision problems, e.g.
  - Decide if an equation is valid for an equational theory,
  - **Decide if two words are equivalent modulo a congruence on words**

The diamond lemma

- The diamond lemma relates confluence and local confluence properties for terminating (abstract) rewriting systems.
  - It provides an effective criterion to check the confluence property: the CP Theorem.
- The CP theorem is formulated in various contexts, e.g.
  - Term rewriting, word rewriting, **Gröbner bases**, $\cdots$
- If each critical pair is confluent, the normal form procedure can be used to solve decision problems, e.g.
  - Decide if an equation is valid for an equational theory,
  - Decide if two words are equivalent modulo a congruence on words,
  - **Decide if a polynomial belongs to a polynomial ideal**

The diamond lemma

- The diamond lemma relates confluence and local confluence properties for terminating (abstract) rewriting systems.
  - It provides an effective criterion to check the confluence property: the CP Theorem.
- The CP theorem is formulated in various contexts, e.g.
  - Term rewriting, word rewriting, Gröbner bases, $\cdots$
- If each critical pair is confluent, the normal form procedure can be used to solve decision problems, e.g.
  - Decide if an equation is valid for an equational theory,
  - Decide if two words are equivalent modulo a congruence on words,
  - Decide if a polynomial belongs to a polynomial ideal,
  - $\cdots$

Rewriting theory and functional systems

- Gröbner bases provide symbolic methods for studying properties of algebras over a field $\mathbb{K}$.
  - e.g., computation of Hilbert series, computation of homological invariants, $\cdots$

Rewriting theory and functional systems

- Gröbner bases provide symbolic methods for studying properties of algebras over a field $\mathbb{K}$.
  - e.g., computation of Hilbert series, computation of homological invariants, $\cdots$
  - These methods require to compute canonical representatives using normal forms.

Rewriting theory and functional systems

- Gröbner bases provide symbolic methods for studying properties of algebras over a field $\mathbb{K}$.
  - e.g., computation of Hilbert series, computation of homological invariants, $\cdots$
  - These methods require to compute canonical representatives using normal forms.
  - The confluence property can be shown using CP.

Rewriting theory and functional systems

- Gröbner bases provide symbolic methods for studying properties of algebras over a field $\mathbb{K}$.
  - e.g., computation of Hilbert series, computation of homological invariants, $\cdots$
  - These methods require to compute canonical representatives using normal forms.
  - The confluence property can be shown using CP.
- Consider an algebra of operators **A**.

Rewriting theory and functional systems

- Gröbner bases provide symbolic methods for studying properties of algebras over a field $\mathbb{K}$.
  - e.g., computation of Hilbert series, computation of homological invariants, $\cdots$
  - These methods require to compute canonical representatives using normal forms.
  - The confluence property can be shown using CP.
- Consider an algebra of operators $\mathbf{A}$.
  - e.g., $\mathbf{A} = C^\infty(\mathbb{R}^n)[\partial_1, \cdots, \partial_n]$ is a model for PDE systems.

Rewriting theory and functional systems

- Gröbner bases provide symbolic methods for studying properties of algebras over a field $\mathbb{K}$.
  - e.g., computation of Hilbert series, computation of homological invariants, $\cdots$
  - These methods require to compute canonical representatives using normal forms.
  - The confluence property can be shown using CP.
- Consider an algebra of operators $\mathbf{A}$.
  - e.g., $\mathbf{A} = C^\infty(\mathbb{R}^n)[\partial_1, \cdots, \partial_n]$ is a model for PDE systems.
  - Properties of $\mathbf{A}$ can be used to study functional systems.

Rewriting theory and functional systems

- Gröbner bases provide symbolic methods for studying properties of algebras over a field $\mathbb{K}$.
  - e.g., computation of Hilbert series, computation of homological invariants, $\cdots$
  - These methods require to compute canonical representatives using normal forms.
  - The confluence property can be shown using CP.
- Consider an algebra of operators $\mathbf{A}$.
  - e.g., $\mathbf{A} = C^\infty(\mathbb{R}^n)[\partial_1, \cdots, \partial_n]$ is a model for PDE systems.
  - Properties of $\mathbf{A}$ can be used to study functional systems.
  - How to prove that a rewriting system over $\mathbf{A}$ is confluent?

Rewriting theory and functional systems

- Gröbner bases provide symbolic methods for studying properties of algebras over a field $\mathbb{K}$.
  - e.g., computation of Hilbert series, computation of homological invariants, $\cdots$
  - These methods require to compute canonical representatives using normal forms.
  - The confluence property can be shown using CP.
- Consider an algebra of operators $\mathbf{A}$.
  - e.g., $\mathbf{A} = C^\infty(\mathbb{R}^n)[\partial_1, \cdots, \partial_n]$ is a model for PDE systems.
  - Properties of $\mathbf{A}$ can be used to study functional systems.
  - How to prove that a rewriting system over $\mathbf{A}$ is confluent?
- Preliminary: how to decide if a (terminating) rewriting system over a **free module** $\mathbf{R}X$ is locally confluent?

Rewriting theory and functional systems

- Gröbner bases provide symbolic methods for studying properties of algebras over a field $\mathbb{K}$.
  - e.g., computation of Hilbert series, computation of homological invariants, $\cdots$
  - These methods require to compute canonical representatives using normal forms.
  - The confluence property can be shown using CP.
- Consider an algebra of operators $\mathbf{A}$.
  - e.g., $\mathbf{A} = C^\infty(\mathbb{R}^n)[\partial_1, \cdots, \partial_n]$ is a model for PDE systems.
  - Properties of $\mathbf{A}$ can be used to study functional systems.
  - How to prove that a rewriting system over $\mathbf{A}$ is confluent?
- Preliminary: how to decide if a (terminating) rewriting system over a free module $\mathbf{R}X$ is locally confluent?
  - $\mathbf{R}X$ is the set of finite sums $\sum r_x x$, where $x \in X$ and $r_x \in \mathbf{R}$.

Rewriting theory and functional systems

- Gröbner bases provide symbolic methods for studying properties of algebras over a field $\mathbb{K}$.
  - e.g., computation of Hilbert series, computation of homological invariants, $\cdots$
  - These methods require to compute canonical representatives using normal forms.
  - The confluence property can be shown using CP.
- Consider an algebra of operators $\mathbf{A}$.
  - e.g., $\mathbf{A} = C^\infty(\mathbb{R}^n)[\partial_1, \cdots, \partial_n]$ is a model for PDE systems.
  - Properties of $\mathbf{A}$ can be used to study functional systems.
  - How to prove that a rewriting system over $\mathbf{A}$ is confluent?
- Preliminary: how to decide if a (terminating) rewriting system over a free module $\mathbf{R}X$ is locally confluent?
  - $\mathbf{R}X$ is the set of finite sums $\sum r_x x$, where $x \in X$ and $r_x \in \mathbf{R}$.
  - Example: $\mathbf{R} = C^\infty(\mathbb{R}^n)$ and $X = \left\{ \partial_1^{k_1} \cdots \partial_n^{k_n} \mid k_i \in \mathbb{N} \right\}$.

Objective

- A rewriting system over a vector space is locally confluent iff each basis element is UNF.

Objective

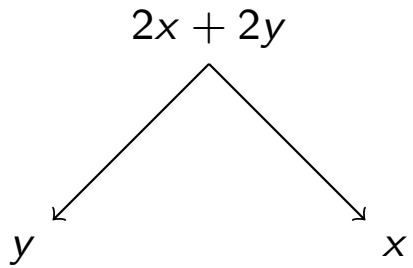
- A rewriting system over a vector space is locally confluent iff each basis element is UNF.
  - For algebras over a field: (diamond lemma $\wedge$ CP lemma) $\Longrightarrow$ CP Theorem.

Objective

- A rewriting system over a vector space is locally confluent iff each basis element is UNF.
  - For algebras over a field: (diamond lemma $\wedge$ CP lemma) $\Longrightarrow$ CP Theorem.
- Consider the rewrite rules $2x \to -y$ and $2y \to -x$ over $\mathbb{Z}\{x,\ y\}$.

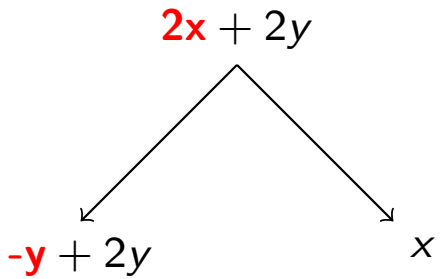
Objective

- A rewriting system over a vector space is locally confluent iff each basis element is UNF.
  - For algebras over a field: (diamond lemma $\wedge$ CP lemma) $\Longrightarrow$ CP Theorem.
- Consider the rewrite rules $2x \to -y$ and $2y \to -x$ over $\mathbb{Z}\{x,\ y\}$.
  - The induced rewrite relation is not confluent since



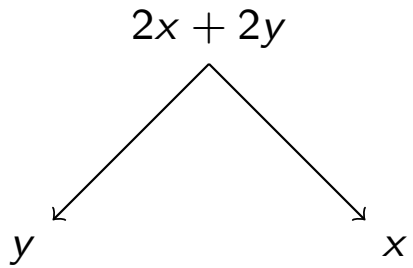
Objective

- A rewriting system over a vector space is locally confluent iff each basis element is UNF.
  - For algebras over a field: (diamond lemma $\wedge$ CP lemma) $\Longrightarrow$ CP Theorem.
- Consider the rewrite rules **2x** $\rightarrow$ **-y** and $2y \rightarrow -x$ over $\mathbb{Z}\{x,\ y\}$.
  - The induced rewrite relation is not confluent since



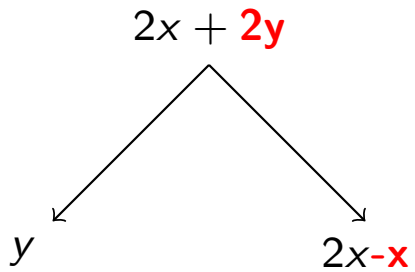
Objective

- A rewriting system over a vector space is locally confluent iff each basis element is UNF.
  - For algebras over a field: (diamond lemma $\wedge$ CP lemma) $\Longrightarrow$ CP Theorem.
- Consider the rewrite rules $2x \to -y$ and $2y \to -x$ over $\mathbb{Z}\{x,\ y\}$.
  - The induced rewrite relation is not confluent since



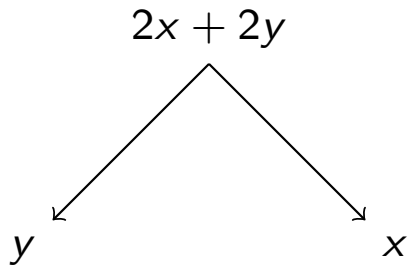
Objective

- A rewriting system over a vector space is locally confluent iff each basis element is UNF.
  - For algebras over a field: (diamond lemma $\wedge$ CP lemma) $\Longrightarrow$ CP Theorem.
- Consider the rewrite rules $2x \to -y$ and **2y** $\to$ **-x** over $\mathbb{Z}\{x,\ y\}$.
  - The induced rewrite relation is not confluent since



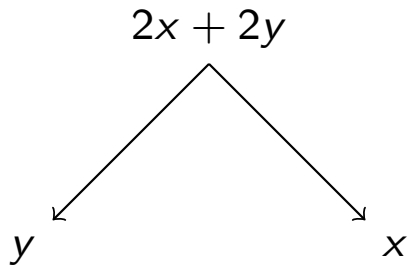
Objective

- A rewriting system over a vector space is locally confluent iff each basis element is UNF.
  - For algebras over a field: (diamond lemma $\wedge$ CP lemma) $\Longrightarrow$ CP Theorem.
- Consider the rewrite rules $2x \to -y$ and $2y \to -x$ over $\mathbb{Z}\{x,\ y\}$.
  - The induced rewrite relation is not confluent since



Objective

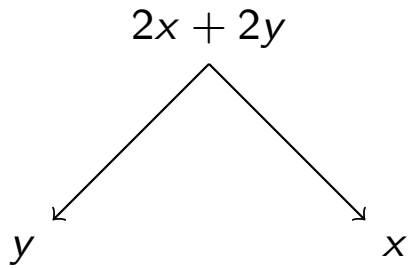
- A rewriting system over a vector space is locally confluent iff each basis element is UNF.
  - For algebras over a field: (diamond lemma $\wedge$ CP lemma) $\Longrightarrow$ CP Theorem.
- Consider the rewrite rules $2x \to -y$ and $2y \to -x$ over $\mathbb{Z}\{x,\ y\}$.
  - The induced rewrite relation is not confluent since



  - However, we will see that both x and y are normal forms!

Objective

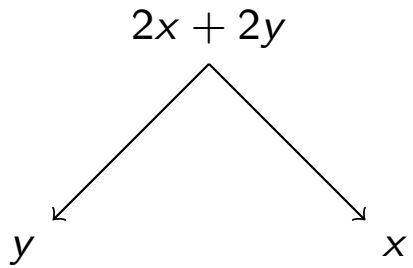
- A rewriting system over a vector space is locally confluent iff each basis element is UNF.
  - For algebras over a field: (diamond lemma $\wedge$ CP lemma) $\Longrightarrow$ CP Theorem.
- Consider the rewrite rules $2x \to -y$ and $2y \to -x$ over $\mathbb{Z}\{x,\ y\}$.
  - The induced rewrite relation is not confluent since



  - However, we will see that both x and y are normal forms!
- Our objective: determine the possible obstructions to the local confluence property.

Objective

- A rewriting system over a vector space is locally confluent iff each basis element is UNF.
  - For algebras over a field: (diamond lemma $\wedge$ CP lemma) $\Longrightarrow$ CP Theorem.
- Consider the rewrite rules $2x \to -y$ and $2y \to -x$ over $\mathbb{Z}\{x,\ y\}$.
  - The induced rewrite relation is not confluent since



  - However, we will see that both x and y are normal forms!
- Our objective: determine the possible obstructions to the local confluence property.
  - We do not deal with critical pairs!

Plan

II. Rewriting over free modules

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed **left-monomial** set of rewrite rules over $\mathbf{R}X$

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to f$

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $\mathbf{r}\mathbf{x} \to f$, where $\mathbf{r} \in \mathbf{R}$, $\mathbf{x} \in \mathbf{X}$

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to \mathbf{f}$, where $r \in \mathbf{R}$, $x \in X$ and $\mathbf{f} \in \mathbf{RX}$.

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to f$, where $r \in \mathbf{R}$, $x \in X$ and $f \in \mathbf{R}X$.
- How to extend $\mathscr{R}$ into a rewriting system over $\mathbf{R}X$?
  - $\forall rx \to f$, choose representatives of left classes modulo r.

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to f$, where $r \in \mathbf{R}$, $x \in X$ and $f \in \mathbf{R}X$.
- How to extend $\mathscr{R}$ into a rewriting system over $\mathbf{R}X$?
  - $\forall rx \to f$, choose representatives of left classes modulo r.
  - Each $s \in \mathbf{R}$ admits a decomposition $s'r + \bar{s}$, where $\bar{s}$ is the representative of s.

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to f$, where $r \in \mathbf{R}$, $x \in X$ and $f \in \mathbf{R}X$.
- How to extend $\mathscr{R}$ into a rewriting system over $\mathbf{R}X$?
  - $\forall rx \to f$, choose representatives of left classes modulo r.
  - Each $s \in \mathbf{R}$ admits a decomposition $s'r + \bar{s}$, where $\bar{s}$ is the representative of s.
  - We let

$$sx + \sum r_y y \to s'f + \bar{s}x + \sum r_y y,$$

whenever each y is different from x.

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to f$, where $r \in \mathbf{R}$, $x \in X$ and $f \in \mathbf{R}X$.
- How to extend $\mathscr{R}$ into a rewriting system over $\mathbf{R}X$?
  - $\forall rx \to f$, choose representatives of left classes modulo r.
  - Each $s \in \mathbf{R}$ admits a decomposition $\mathbf{s'r} + \bar{\mathbf{s}}$, where $\bar{s}$ is the representative of s.
  - We let

$$\mathbf{s}x + \sum r_y y \;=\; \mathbf{s'r}x \;+\; \bar{\mathbf{s}}x + \sum r_y y \to s'f + \bar{s}x + \sum r_y y,$$

whenever each y is different from x.

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to f$, where $r \in \mathbf{R}$, $x \in X$ and $f \in \mathbf{R}X$.
- How to extend $\mathscr{R}$ into a rewriting system over $\mathbf{R}X$?
  - $\forall \mathbf{rx} \to \mathbf{f}$, choose representatives of left classes modulo r.
  - Each $s \in \mathbf{R}$ admits a decomposition $s'r + \bar{s}$, where $\bar{s}$ is the representative of s.
  - We let

$$sx + \sum r_y y \;=\; s'\mathbf{rx} \;+\; \bar{s}x + \sum r_y y \to s'\mathbf{f} + \bar{s}x + \sum r_y y,$$

whenever each y is different from x.

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to f$, where $r \in \mathbf{R}$, $x \in X$ and $f \in \mathbf{R}X$.
- How to extend $\mathscr{R}$ into a rewriting system over $\mathbf{R}X$?
  - $\forall rx \to f$, choose representatives of left classes modulo r.
  - Each $s \in \mathbf{R}$ admits a decomposition $s'r + \bar{s}$, where $\bar{s}$ is the representative of s.
  - We let

$$sx + \sum r_y y \to s'f + \bar{s}x + \sum r_y y,$$

whenever each y is different from x.

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to f$, where $r \in \mathbf{R}$, $x \in X$ and $f \in \mathbf{R}X$.
- How to extend $\mathscr{R}$ into a rewriting system over $\mathbf{R}X$?
  - $\forall rx \to f$, choose representatives of left classes modulo r.
  - Each $s \in \mathbf{R}$ admits a decomposition $s'r + \bar{s}$, where $\bar{s}$ is the representative of s.
  - We let

$$sx + \sum r_y y \to s'f + \bar{s}x + \sum r_y y,$$

    whenever each y is different from x.
- Example: if $\mathbf{R} = \mathbb{Z}$, canonical representatives are reminders for euclidean division.

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to f$, where $r \in \mathbf{R}$, $x \in X$ and $f \in \mathbf{R}X$.
- How to extend $\mathscr{R}$ into a rewriting system over $\mathbf{R}X$?
  - $\forall rx \to f$, choose representatives of left classes modulo r.
  - Each $s \in \mathbf{R}$ admits a decomposition $s'r + \bar{s}$, where $\bar{s}$ is the representative of s.
  - We let

$$sx + \sum r_y y \to s'f + \bar{s}x + \sum r_y y,$$

    whenever each y is different from x.
- Example: if $\mathbf{R} = \mathbb{Z}$, canonical representatives are reminders for euclidean division.
  - If $\mathscr{R} = \{2x \to -y,\ 2y \to -x\}$, then x and y are normal forms.

Rewrite rules over free modules

- $\mathscr{R}$ is a fixed left-monomial set of rewrite rules over $\mathbf{R}X$, that is
  - elements of $\mathscr{R}$ are of the form $rx \to f$, where $r \in \mathbf{R}$, $x \in X$ and $f \in \mathbf{R}X$.
- How to extend $\mathscr{R}$ into a rewriting system over $\mathbf{R}X$?
  - $\forall rx \to f$, choose representatives of left classes modulo r.
  - Each $s \in \mathbf{R}$ admits a decomposition $s'r + \bar{s}$, where $\bar{s}$ is the representative of s.
  - We let

$$sx + \sum r_y y \to s'f + \bar{s}x + \sum r_y y,$$

    whenever each y is different from x.
- Example: if $\mathbf{R} = \mathbb{Z}$, canonical representatives are reminders for euclidean division.
  - If $\mathscr{R} = \{2x \to -y,\ 2y \to -x\}$, then x and y are normal forms.

Proposition. *The equational theory of $\mathscr{R}$ is the module generated by the elements $rx - f$.*

An example: the Ore extension

- Let σ be a **ring endomorphism**

An example: the Ore extension

- Let σ be a ring endomorphism
  - $\sigma(0) = 0$, $\sigma(1) = 1$, $\sigma(r + r') = \sigma(r) + \sigma(r')$ and $\sigma(rr') = \sigma(r)\sigma(r')$, $\forall r, r' \in \mathbf{R}$

An example: the Ore extension

- Let σ be a ring endomorphism and δ be a $\sigma-$**derivation**, that is
  - $\sigma(0) = 0$, $\sigma(1) = 1$, $\sigma(r + r') = \sigma(r) + \sigma(r')$ and $\sigma(rr') = \sigma(r)\sigma(r')$, $\forall r, r' \in \mathbf{R}$

An example: the Ore extension

- Let σ be a ring endomorphism and δ be a σ-derivation, that is
  - $\sigma(0) = 0$, $\sigma(1) = 1$, $\sigma(r + r') = \sigma(r) + \sigma(r')$ and $\sigma(rr') = \sigma(r)\sigma(r')$, $\forall r, r' \in \mathbf{R}$,
  - $\delta(r + r') = \delta(r) + \delta(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.

An example: the Ore extension

- Let σ be a ring endomorphism and δ be a σ-derivation, that is
  - $\sigma(0) = 0$, $\sigma(1) = 1$, $\sigma(r + r') = \sigma(r) + \sigma(r')$ and $\sigma(rr') = \sigma(r)\sigma(r')$, $\forall r, r' \in \mathbf{R}$,
  - $\delta(r + r') = \delta(r) + \delta(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- The **Ore extension** of $\mathbf{R}$ is the skew-polynomial ring $\mathbf{R}[\partial]$

An example: the Ore extension

- Let σ be a ring endomorphism and δ be a σ-derivation, that is
  - $\sigma(0) = 0$, $\sigma(1) = 1$, $\sigma(r + r') = \sigma(r) + \sigma(r')$ and $\sigma(rr') = \sigma(r)\sigma(r')$, $\forall r, r' \in \mathbf{R}$,
  - $\delta(r + r') = \delta(r) + \delta(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- The Ore extrension of $\mathbf{R}$ is the skew-polynomial ring $\mathbf{R}[\partial]$:
  - the underlying set is composed of polynomials in ∂: $r_0 + r_1\partial + \cdots + r_n\partial^n$,

An example: the Ore extension

- Let σ be a ring endomorphism and δ be a σ-derivation, that is
  - $\sigma(0) = 0$, $\sigma(1) = 1$, $\sigma(r + r') = \sigma(r) + \sigma(r')$ and $\sigma(rr') = \sigma(r)\sigma(r')$, $\forall r, r' \in \mathbf{R}$,
  - $\delta(r + r') = \delta(r) + \delta(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- The Ore extrension of $\mathbf{R}$ is the skew-polynomial ring $\mathbf{R}[\partial]$:
  - the underlying set is composed of polynomials in ∂: $r_0 + r_1\partial + \cdots + r_n\partial^n$,
  - the right product of $r \in \mathbf{R}$ on ∂ is given by: $\partial r = \sigma(r)\partial + \delta(r)$.

An example: the Ore extension

- Let σ be a ring endomorphism and δ be a σ-derivation, that is
  - $\sigma(0) = 0$, $\sigma(1) = 1$, $\sigma(r + r') = \sigma(r) + \sigma(r')$ and $\sigma(rr') = \sigma(r)\sigma(r')$, $\forall r, r' \in \mathbf{R}$,
  - $\delta(r + r') = \delta(r) + \delta(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- The Ore extrension of $\mathbf{R}$ is the skew-polynomial ring $\mathbf{R}[\partial]$:
  - the underlying set is composed of polynomials in ∂: $r_0 + r_1\partial + \cdots + r_n\partial^n$,
  - the right product of $r \in \mathbf{R}$ on ∂ is given by: $\partial r = \sigma(r)\partial + \delta(r)$.
- In degrees 0 and 1

An example: the Ore extension

- Let σ be a ring endomorphism and δ be a σ-derivation, that is
  - $\sigma(0) = 0$, $\sigma(1) = 1$, $\sigma(r + r') = \sigma(r) + \sigma(r')$ and $\sigma(rr') = \sigma(r)\sigma(r')$, $\forall r, r' \in \mathbf{R}$,
  - $\delta(r + r') = \delta(r) + \delta(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- The Ore extrension of $\mathbf{R}$ is the skew-polynomial ring $\mathbf{R}[\partial]$:
  - the underlying set is composed of polynomials in ∂: $r_0 + r_1\partial + \cdots + r_n\partial^n$,
  - the right product of $r \in \mathbf{R}$ on ∂ is given by: $\partial r = \sigma(r)\partial + \delta(r)$.
- In degrees 0 and 1:
  - $\mathbf{R}[\partial]$ is a quotient of the free module $\mathbf{R}\{1,\ \partial\}$.

An example: the Ore extension

- Let σ be a ring endomorphism and δ be a σ-derivation, that is
  - $\sigma(0) = 0$, $\sigma(1) = 1$, $\sigma(r + r') = \sigma(r) + \sigma(r')$ and $\sigma(rr') = \sigma(r)\sigma(r')$, $\forall r, r' \in \mathbf{R}$,
  - $\delta(r + r') = \delta(r) + \delta(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- The Ore extrension of $\mathbf{R}$ is the skew-polynomial ring $\mathbf{R}[\partial]$:
  - the underlying set is composed of polynomials in ∂: $r_0 + r_1\partial + \cdots + r_n\partial^n$,
  - the right product of $r \in \mathbf{R}$ on ∂ is given by: $\partial r = \sigma(r)\partial + \delta(r)$.
- In degrees 0 and 1:
  - $\mathbf{R}[\partial]$ is a quotient of the free module $\mathbf{R}\{1,\ \partial\}$.
  - This quotient is the equational theory of

$$\mathscr{R} = \left\{ \partial r \longrightarrow \sigma(r)\partial + \delta(r) \ \mid \ r \in \mathbf{R} \right\}.$$

An example: the Ore extension

- Let σ be a ring endomorphism and δ be a σ-derivation, that is
  - $\sigma(0) = 0$, $\sigma(1) = 1$, $\sigma(r + r') = \sigma(r) + \sigma(r')$ and $\sigma(rr') = \sigma(r)\sigma(r')$, $\forall r, r' \in \mathbf{R}$,
  - $\delta(r + r') = \delta(r) + \delta(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- The Ore extrension of $\mathbf{R}$ is the skew-polynomial ring $\mathbf{R}[\partial]$:
  - the underlying set is composed of polynomials in ∂: $r_0 + r_1\partial + \cdots + r_n\partial^n$,
  - the right product of $r \in \mathbf{R}$ on ∂ is given by: $\partial r = \sigma(r)\partial + \delta(r)$.
- In degrees 0 and 1:
  - $\mathbf{R}[\partial]$ is a quotient of the free module $\mathbf{R}\{1,\ \partial\}$.
  - This quotient is the equational theory of

$$\mathscr{R} = \left\{ \partial r \longrightarrow \sigma(r)\partial + \delta(r) \ \mid \ r \in \mathbf{R} \right\}.$$

  - Is the rewriting system induced by $\mathscr{R}$ confluent?

Plan

III. The diamond lemma

Syzygies

- We fix a set of rewrite rules $\mathscr{R}$ over $\mathbf{R}X$.

Syzygies

- We fix a set of rewrite rules $\mathscr{R}$ over $\mathbf{R}X$.
  - How to determine the possible obstructions to confluence?

Syzygies

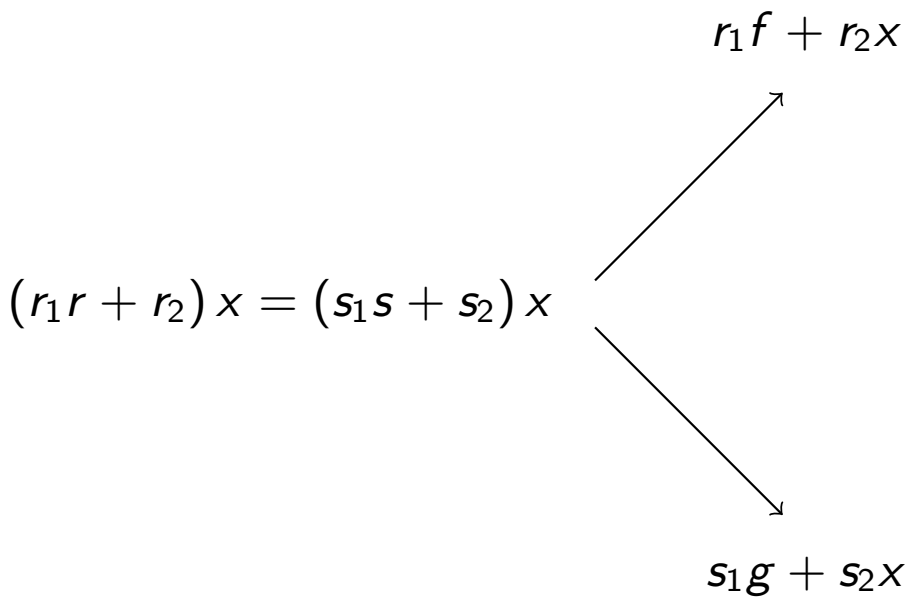
- We fix a set of rewrite rules $\mathscr{R}$ over $\mathbf{R}X$.
  - How to determine the possible obstructions to confluence?
- A **syzygy** of a pair of rewrite rules $(rx \to f,\ sx \to g)$ is a tuple (r_1, r_2, s_2, s_2) s.t.
  - r_2 (resp. s_2) is the representative of its left class modulo r (resp. s),
  - $r_1 r + r_2 = s_1 s + s_2$.

Syzygies

- We fix a set of rewrite rules $\mathscr{R}$ over $\mathbf{R}X$.
  - How to determine the possible obstructions to confluence?
- A syzygy of a pair of rewrite rules $(rx \to f,\ sx \to g)$ is a tuple (r_1, r_2, s_2, s_2) s.t.
  - r_2 (resp. s_2) is the representative of its left class modulo r (resp. s),
  - $r_1 r + r_2 = s_1 s + s_2$.
- The syzygy (r_1, r_2, s_2, s_2) is said to be **confluent** if

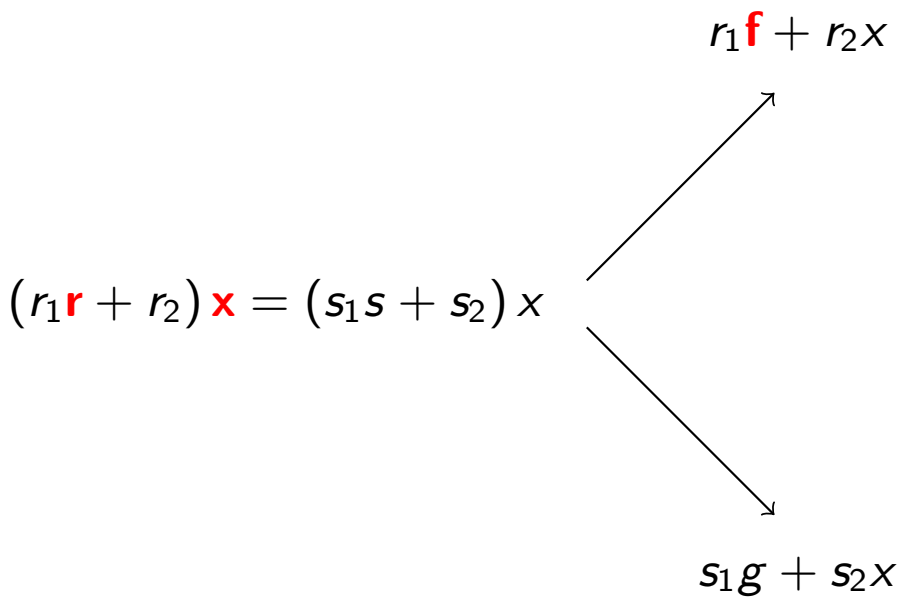
Syzygies

- We fix a set of rewrite rules $\mathscr{R}$ over $\mathbf{R}X$.
  - How to determine the possible obstructions to confluence?
- A syzygy of a pair of rewrite rules $(rx \to f,\ sx \to g)$ is a tuple (r_1, r_2, s_2, s_2) s.t.
  - r_2 (resp. s_2) is the representative of its left class modulo r (resp. s),
  - $r_1 r + r_2 = s_1 s + s_2$.
- The syzygy (r_1, r_2, s_2, s_2) is said to be **confluent** if



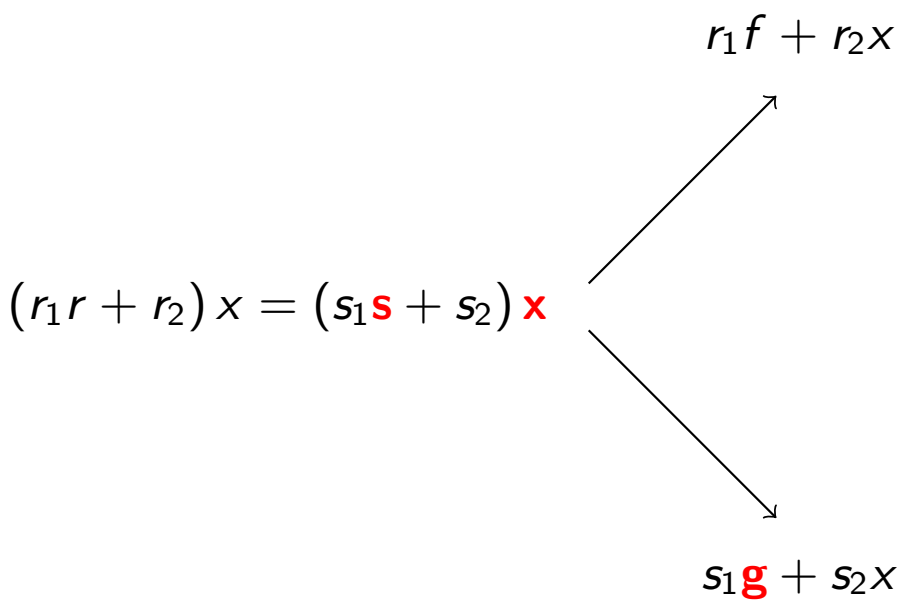
Syzygies

- We fix a set of rewrite rules $\mathscr{R}$ over $\mathbf{R}X$.
  - How to determine the possible obstructions to confluence?
- A syzygy of a pair of rewrite rules $(rx \to f,\ sx \to g)$ is a tuple (r_1, r_2, s_2, s_2) s.t.
  - r_2 (resp. s_2) is the representative of its left class modulo r (resp. s),
  - $r_1 r + r_2 = s_1 s + s_2$.
- The syzygy (r_1, r_2, s_2, s_2) is said to be confluent if



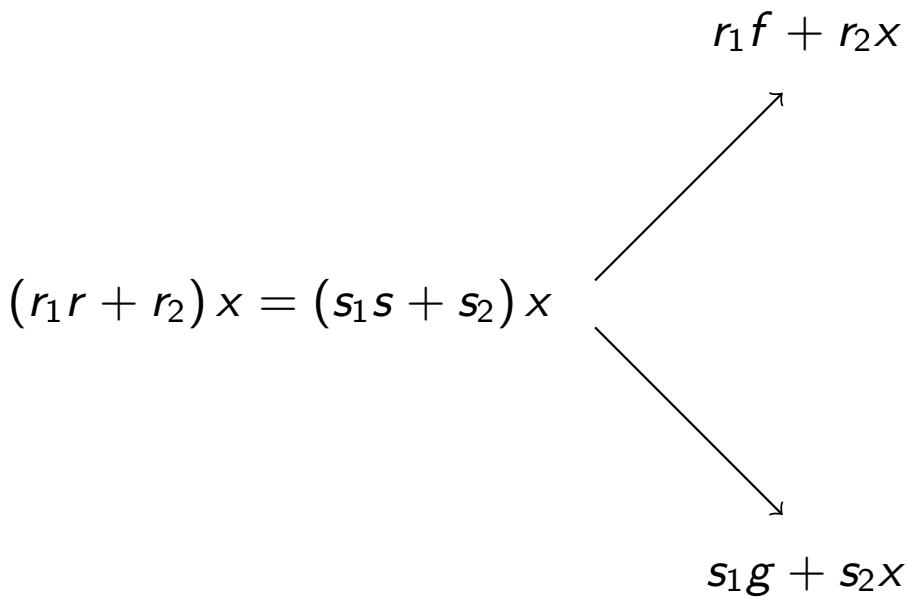
Syzygies

- We fix a set of rewrite rules $\mathscr{R}$ over $\mathbf{R}X$.
  - How to determine the possible obstructions to confluence?
- A syzygy of a pair of rewrite rules $(rx \to f,\ sx \to g)$ is a tuple (r_1, r_2, s_2, s_2) s.t.
  - r_2 (resp. s_2) is the representative of its left class modulo r (resp. s),
  - $r_1 r + r_2 = s_1 s + s_2$.
- The syzygy (r_1, r_2, s_2, s_2) is said to be confluent if



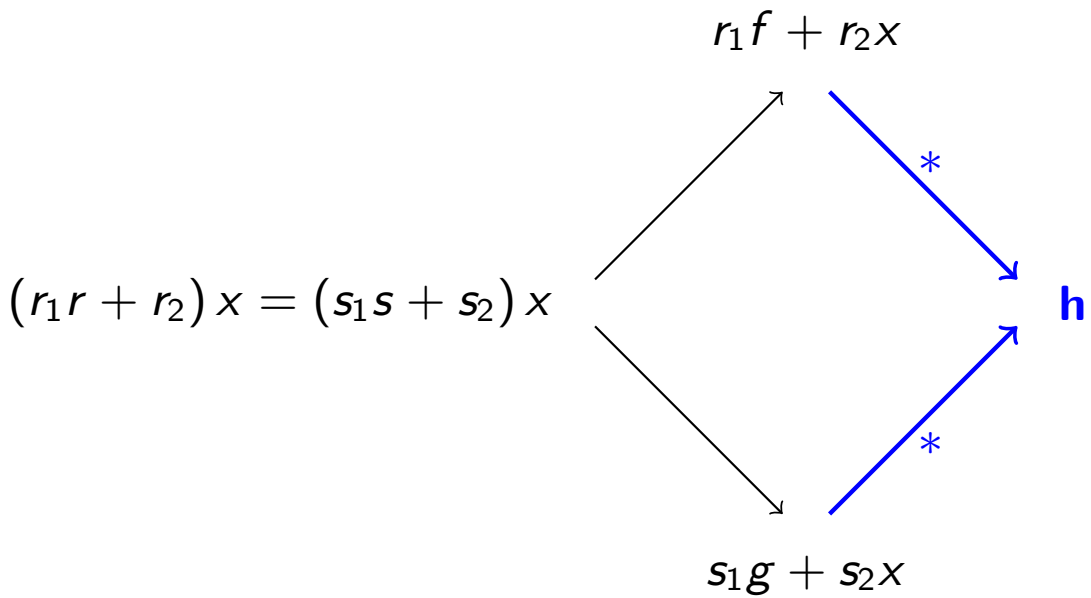
Syzygies

- We fix a set of rewrite rules $\mathscr{R}$ over $\mathbf{R}X$.
  - How to determine the possible obstructions to confluence?
- A syzygy of a pair of rewrite rules $(rx \to f,\ sx \to g)$ is a tuple (r_1, r_2, s_2, s_2) s.t.
  - r_2 (resp. s_2) is the representative of its left class modulo r (resp. s),
  - $r_1 r + r_2 = s_1 s + s_2$.
- The syzygy (r_1, r_2, s_2, s_2) is said to be **confluent** if



Syzygies

- We fix a set of rewrite rules $\mathscr{R}$ over $\mathbf{R}X$.
  - How to determine the possible obstructions to confluence?
- A syzygy of a pair of rewrite rules $(rx \to f,\ sx \to g)$ is a tuple (r_1, r_2, s_2, s_2) s.t.
  - r_2 (resp. s_2) is the representative of its left class modulo r (resp. s),
  - $r_1 r + r_2 = s_1 s + s_2$.
- The syzygy (r_1, r_2, s_2, s_2) is said to be **confluent** if



The criterion

- The **additivity condition** for $\mathscr{R}$ is
  - $\forall\ (rx \to f,\ h) \in \mathscr{R} \times \mathbf{R}X$, we have $rx + h \downarrow f + h$.

The criterion

- The additivity condition for $\mathscr{R}$ is
  - $\forall\ (rx \to f,\ h) \in \mathscr{R} \times \mathbf{R}X$, we have $rx + h \downarrow f + h$.
- A **compatible termination order** for $\mathscr{R}$ is a partial order $\preceq$ over $\mathbf{R}X$ such that
  - $\preceq$ is well-founded,
  - $f \to g \Rightarrow g \prec f$,
  - $\preceq$ is "compatible" with the module structure.

The criterion

- The additivity condition for $\mathscr{R}$ is
  - $\forall\ (rx \to f,\ h) \in \mathscr{R} \times \mathbf{R}X$, we have $rx + h \downarrow f + h$.

- A compatible termination order for $\mathscr{R}$ is a partial order $\preceq$ over $\mathbf{R}X$ such that
  - $\preceq$ is well-founded,
  - $f \to g \Rightarrow g \prec f$,
  - $\preceq$ is "compatible" with the module structure.

Theorem. *Assume that $\mathscr{R}$ satisfies the additivity condition and let $\preceq$ being a compatible termination order for $\mathscr{R}$. The rewriting system induced by $\mathscr{R}$ is locally confluent if and only if every syzygy is confluent.*

The criterion

- The additivity condition for $\mathscr{R}$ is
  - $\forall\ (rx \to f,\ h) \in \mathscr{R} \times \mathbf{R}X$, we have $rx + h \downarrow f + h$.

- A compatible termination order for $\mathscr{R}$ is a partial order $\preceq$ over $\mathbf{R}X$ such that
  - $\preceq$ is well-founded,
  - $f \to g \Rightarrow g \prec f$,
  - $\preceq$ is "compatible" with the module structure.

Theorem. *Assume that $\mathscr{R}$ satisfies the additivity condition and let $\preceq$ being a compatible termination order for $\mathscr{R}$. The rewriting system induced by $\mathscr{R}$ is locally confluent if and only if every syzygy is confluent.*

Counter-example. $\mathscr{R} = \{2x \to -y,\ 2y \to -x\}$.

The criterion

- The additivity condition for $\mathscr{R}$ is
  - $\forall\ (rx \to f,\ h) \in \mathscr{R} \times \mathbf{R}X$, we have $rx + h \downarrow f + h$.

- A compatible termination order for $\mathscr{R}$ is a partial order $\preceq$ over $\mathbf{R}X$ such that
  - $\preceq$ is well-founded,
  - $f \to g \Rightarrow g \prec f$,
  - $\preceq$ is "compatible" with the module structure.

Theorem. *Assume that $\mathscr{R}$ satisfies the additivity condition and let $\preceq$ being a compatible termination order for $\mathscr{R}$. The rewriting system induced by $\mathscr{R}$ is locally confluent if and only if every syzygy is confluent.*

Counter-example. $\mathscr{R} = \{2x \to -y,\ 2y \to -x\}$.

- $\to$ is terminating.

The criterion

- The additivity condition for $\mathscr{R}$ is
  - $\forall\ (rx \to f,\ h) \in \mathscr{R} \times \mathbf{R}X$, we have $rx + h \downarrow f + h$.

- A compatible termination order for $\mathscr{R}$ is a partial order $\preceq$ over $\mathbf{R}X$ such that
  - $\preceq$ is well-founded,
  - $f \to g \Rightarrow g \prec f$,
  - $\preceq$ is "compatible" with the module structure.

Theorem. *Assume that $\mathscr{R}$ satisfies the additivity condition and let $\preceq$ being a compatible termination order for $\mathscr{R}$. The rewriting system induced by $\mathscr{R}$ is locally confluent if and only if every syzygy is confluent.*

Counter-example. $\mathscr{R} = \{2x \to -y,\ 2y \to -x\}$.

- $\to$ is terminating.
- $\to$ is not compatible with the module structure: $2x + 2y \nrightarrow -y - x$.

Ilustration: the Ore extension

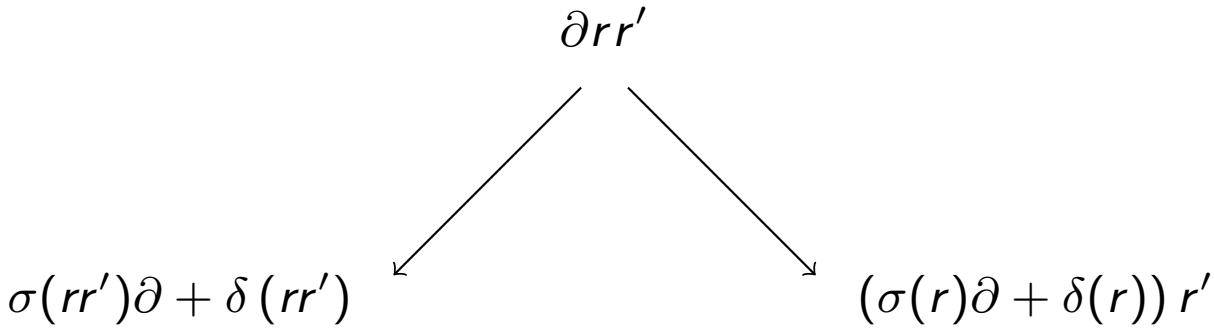
- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.

Ilustration: the Ore extension

- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:

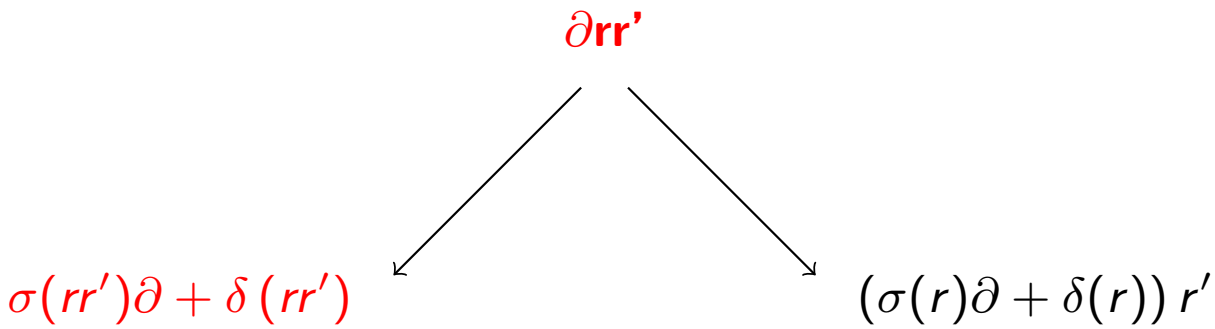
Ilustration: the Ore extension

- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:



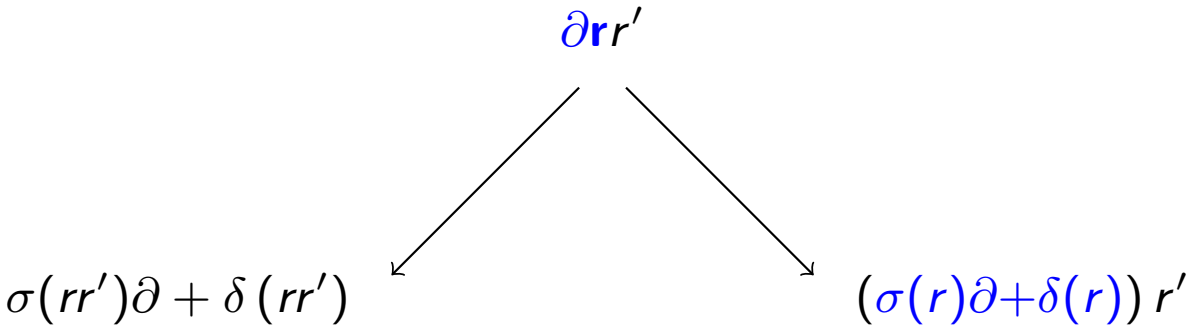
Ilustration: the Ore extension

- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:



Ilustration: the Ore extension

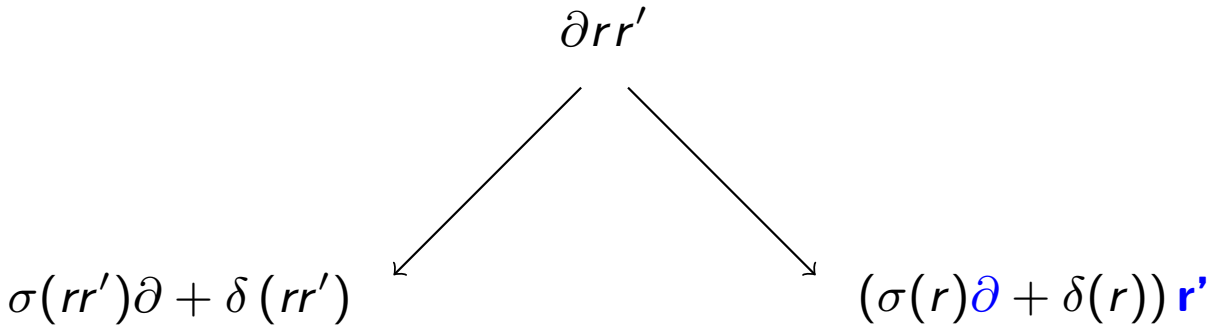
- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:



Ilustration: the Ore extension

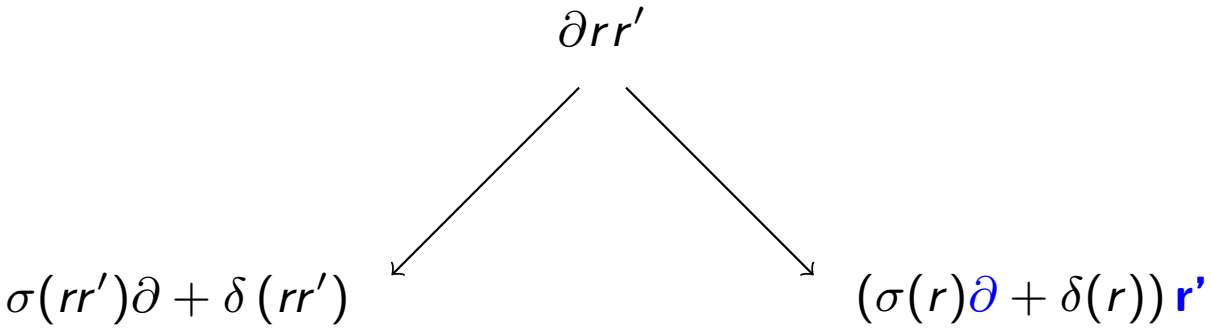
- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.

- An example of syzygy:



Ilustration: the Ore extension

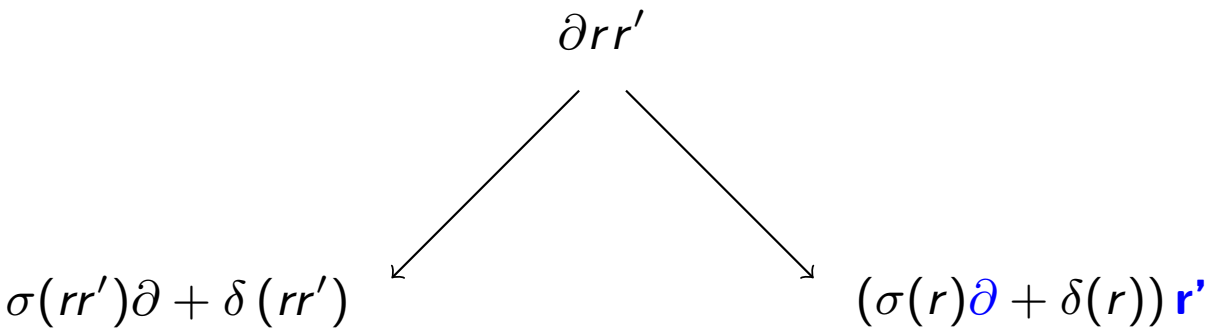
- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:



- $(\sigma(r)\partial + \delta(r))\,\mathbf{r'} \to \sigma(r)\sigma(\mathbf{r'})\partial + \sigma(r)\delta(\mathbf{r'}) + \delta(r)r'$

Ilustration: the Ore extension

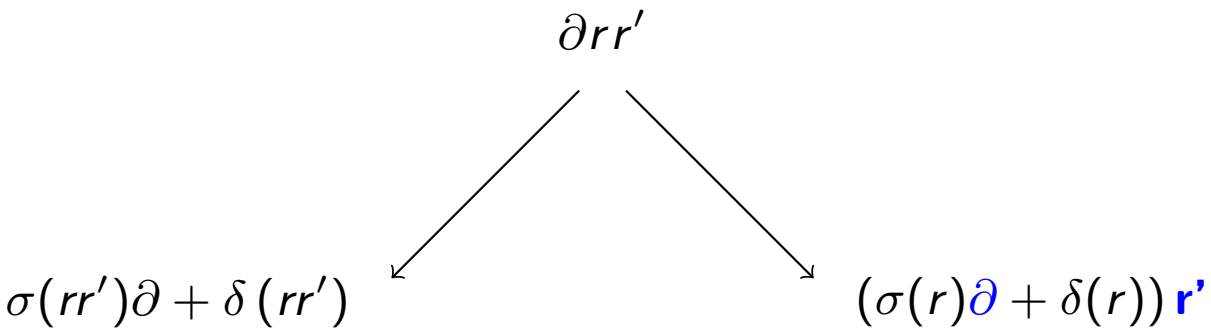
- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:



- $(\sigma(r)\partial + \delta(r))\, r' \to \sigma(r)\sigma(r')\partial + \sigma(r)\delta(r') + \delta(r)r'$

Ilustration: the Ore extension

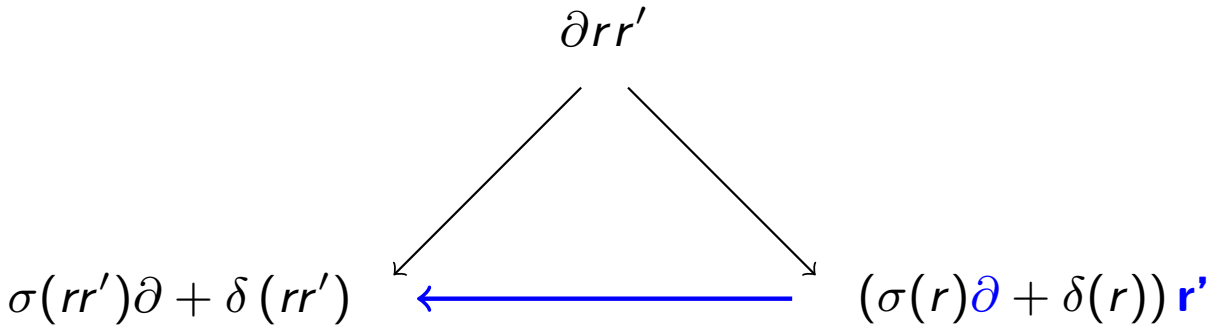
- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:



- $(\sigma(r)\partial + \delta(r))\, r' \to \sigma(r)\sigma(r')\partial + \sigma(r)\delta(r') + \delta(r)r' = \sigma(rr') + \delta(rr')$.

Ilustration: the Ore extension

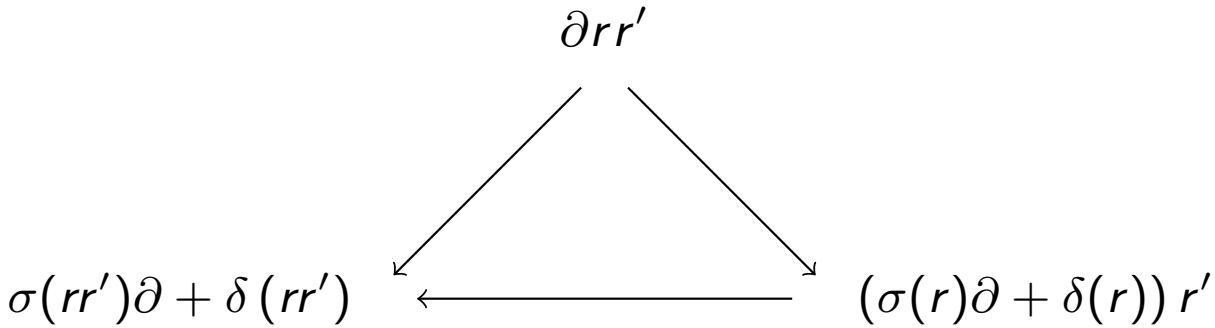
- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:



- $(\sigma(r)\partial + \delta(r))\, r' \to \sigma(r)\sigma(r')\partial + \sigma(r)\delta(r') + \delta(r)r' = \sigma(rr') + \delta(rr')$.

Ilustration: the Ore extension

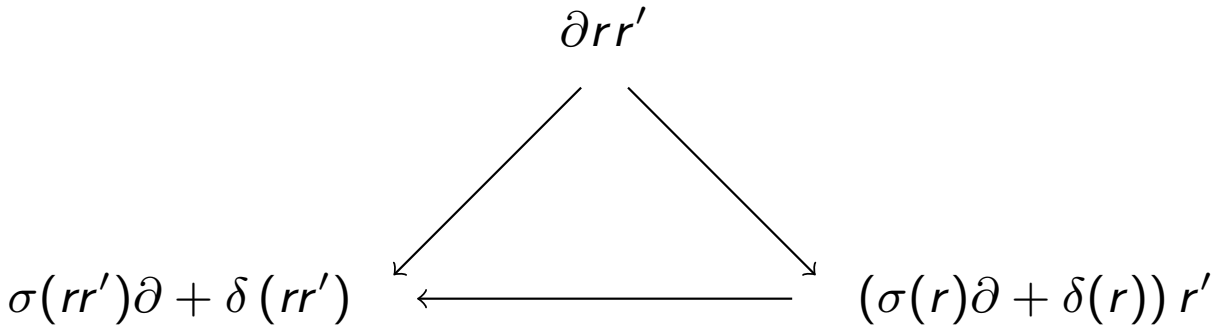
- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:



- $(\sigma(r)\partial + \delta(r))\, r' \to \sigma(r)\sigma(r')\partial + \sigma(r)\delta(r') + \delta(r)r' = \sigma(rr') + \delta(rr')$.
  - In fact, all syzygies are confluent!

Ilustration: the Ore extension

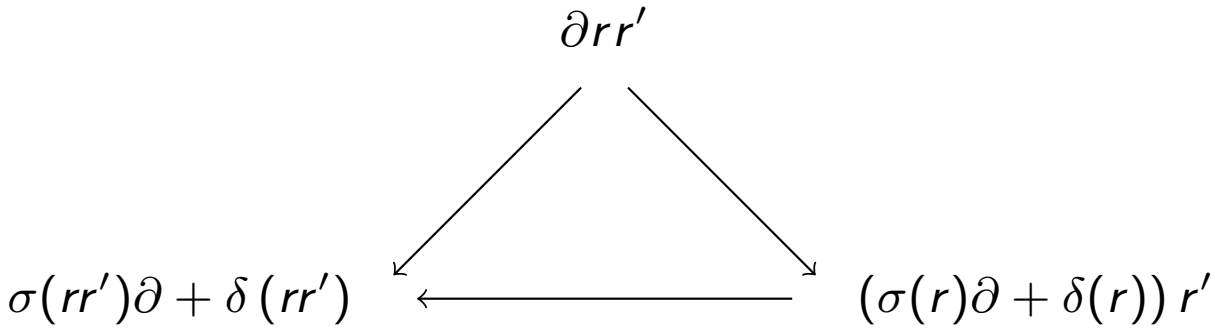
- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:



- $(\sigma(r)\partial + \delta(r))\, r' \to \sigma(r)\sigma(r')\partial + \sigma(r)\delta(r') + \delta(r)r' = \sigma(rr') + \delta(rr')$.
  - In fact, all syzygies are confluent!
- If $\mathbf{R}$ is **left-noetherian**, there is a compatible termination order for $\mathscr{R}$.

Ilustration: the Ore extension

- $\mathscr{R} = \{\partial r \to \sigma(r)\partial + \delta(r) \mid r \in \mathbf{R}\}$, where σ and δ satisfy:
  - $\sigma(rr') = \sigma(r)\sigma(r')$ and $\delta(rr') = \sigma(r)\delta(r') + \delta(r)r'$.
- An example of syzygy:



- $(\sigma(r)\partial + \delta(r))\, r' \to \sigma(r)\sigma(r')\partial + \sigma(r)\delta(r') + \delta(r)r' = \sigma(rr') + \delta(rr')$.
  - In fact, all syzygies are confluent!
- If $\mathbf{R}$ is left-noetherian, there is a compatible termination order for $\mathscr{R}$.
  - The rewrite relation induced by $\mathscr{R}$ is confluent!

Plan

IV. Conclusion and perspectives

Summary and perspectives

- Summary:

Summary and perspectives

- Summary:
  - We introduced rewriting systems over free modules.

Summary and perspectives

- Summary:
  - We introduced rewriting systems over free modules.
  - We obtained a criterion for local confluence in terms of syzygies.

Summary and perspectives

- Summary:
  - We introduced rewriting systems over free modules.
  - We obtained a criterion for local confluence in terms of syzygies.
  - We deduced a convergent presentation of Ore extensions.

Summary and perspectives

- Summary:
  - We introduced rewriting systems over free modules.
  - We obtained a criterion for local confluence in terms of syzygies.
  - We deduced a convergent presentation of Ore extensions.
- Further works:

Summary and perspectives

- Summary:
  - We introduced rewriting systems over free modules.
  - We obtained a criterion for local confluence in terms of syzygies.
  - We deduced a convergent presentation of Ore extensions.
- Further works:
  - Obtain a CP lemma for algebras over (noncommutative) rings and deduce an effective CP theorem.

Summary and perspectives

- Summary:
  - We introduced rewriting systems over free modules.
  - We obtained a criterion for local confluence in terms of syzygies.
  - We deduced a convergent presentation of Ore extensions.
- Further works:
  - Obtain a CP lemma for algebras over (noncommutative) rings and deduce an effective CP theorem.
  - Use rewriting theory to compute algebraic invariants of functional systems.

Summary and perspectives

- Summary:
  - We introduced rewriting systems over free modules.
  - We obtained a criterion for local confluence in terms of syzygies.
  - We deduced a convergent presentation of Ore extensions.
- Further works:
  - Obtain a CP lemma for algebras over (noncommutative) rings and deduce an effective CP theorem.
  - Use rewriting theory to compute algebraic invariants of functional systems.

Thank you for listening!