This part focus on **Operational Semantics**
of formal calculi (and programming languages)
Topics

• Tools to study the operational properties of a system:
  ➢ Rewrite Theory (rewriting=abstract form of program execution)
• Induction and Co-induction proof principles.
• Linear Logic and Proof-Nets.

• Bridging between lambda-calculus and functional programming.
  ➢ Call-by-Value and Call-by Name, weak and lazy calculi.
  ➢ Big-Step and Small-Step operational semantics.
  ➢ Observational equivalence

• Reasoning on programs equivalence:
  ➢ Bisimulation and coinductive methods.

• Beyond pure functional:
  ➢ Probabilistic programming and Bayesian Inference:
    Probabilistic lambda calculi, Bayesian Networks & proof-nets
• **Reference Books:**
  - *D. SANGIORGI: Introduction to Bisimulation and Coinduction* (Cambridge University Press, 2011)

• **Lecture Notes** (by Middeldorp, Laurent, Ong)

  Please send me an email (with LMFI in the subject) to have the lecture notes on Rewriting Theory
Operational semantics
of formal calculi and programming languages

Rewriting theory

- Rewriting = abstract form of program execution
- Paradigmatic example: \( \lambda \)-calculus
  (functional programming language, in its essence)
A colony of chameleons includes 20 red, 18 blue, and 16 green individuals. Whenever two chameleons of different color meet, each changes to the third color. Some time passes during which no chameleons are born or die nor do any enter or leave the colony. Is it possible that at the end of this period, all 54 chameleons are the same color?
Example (Group Theory)

signature

| e (constant) | − (unary, postfix) | · (binary, infix) |

equations

| e · x ≈ x | x− · x ≈ e | (x · y) · z ≈ x · (y · z) |

theorems

| e− ≈E e | (x · y)− ≈E y− · x− |

rewrite rules

| e · x → x | x · e → x |
| x− · x → e | x · x− → e |
| (x · y) · z → x · (y · z) | x−− → x |
| e− → e | (x · y)− → y− · x− |
| x− · (x · y) → y | x · (x− · y) → y |

① s ≈ t is valid in \( \mathcal{E} \) (s ≈E t) if and only if s and t have same \( \mathcal{R} \)-normal form

② \( \mathcal{R} \) admits no infinite computations

① & ② \( \implies \) \( \mathcal{E} \) has decidable validity problem
Example (Combinatory Logic)

signature  S  K  I  (constants)  · (application, binary, infix)

terms  S  ((K · I) · I) · S  (x · z) · (y · z)

rewrite rules  I · x → x
(K · x) · y → x
((S · x) · y) · z → (x · z) · (y · z)

rewriting  ((S · K) · K) · x → (K · x) · (K · x)
           → x

inventor  Moses Schönfinkel (1924)
Example (Lambda Calculus)

signature \( \lambda \) (binds variables) \cdot \texttt{(application), binary, infix)}

terms \( M ::= x \mid (\lambda x. M) \mid (M \cdot M) \)

\(\alpha\) conversion \( \lambda x. x \cdot y \equiv_\alpha \lambda z. z \cdot y \)

\(\beta\) reduction \( (\lambda x. M) \cdot N \to_\beta M[x := N] \)

replace free occurrences of \( x \) in \( M \) by \( N \) (and avoid variable capturing)

rewriting \( (\lambda x. x \cdot x) \cdot (\lambda x. x \cdot x) \to (\lambda x. x \cdot x) \cdot (\lambda x. x \cdot x) \)

inventor Alonzo Church (1932)

both Combinatory Logic and Lambda Calculus are Turing-complete
Operational semantics
of formal calculi and programming languages

Rewriting theory

• Rewriting = abstract form of program execution

• Paradigmatic example: \textit{\lambda-calculus}
  (functional programming language, in its essence)
Rewriting

- **Rewrite Theory** provides a powerful set of tools to study computational and operational properties of a system: normalization, termination, confluence, uniqueness of normal forms.

- Tools to study and compare strategies:
  - Is there a strategy guaranteed to lead to normal form, if any (*normalizing strat.*)?

- **Abstract Rewrite Systems** (ARS) capture the common substratum of rewrite theory (independently from the particular structure of terms) - can be used in the study of any calculus or programming language.
Abstract Rewriting: motivations

**concrete** rewrite formalisms / concrete operational semantics:

- $\lambda$-calculus
- *Quantum/ probabilistic/ non-deterministic/*$\ldots$ $\lambda$-calculus
- Proof-nets / graph rewriting
- Sequent calculus and cut-elimination
- string rewriting
- term rewriting

**abstract** rewriting

- independent from structure of objects that are rewritten
- uniform presentation of properties and proofs
Abstract Rewriting

Basic language
Definition 1.1.1. An abstract rewrite system (ARS for short) is a pair $\mathcal{A} = \langle A, \rightarrow \rangle$ consisting of a set $A$ and a binary relation $\rightarrow$ on $A$. Instead of $(a, b) \in \rightarrow$ we write $a \rightarrow b$ and we say that $a \rightarrow b$ is a rewrite step.

- A (finite) rewrite sequence is a non-empty sequence $(a_0, \ldots, a_n)$ of elements in $A$ such that $a_i \rightarrow a_{i+1}$

  We write $a_0 \rightarrow^n a_n$ or simply $a_0 \rightarrow^* a_n$

- rewrite sequence
  - finite: $a \rightarrow e \rightarrow b \rightarrow c \rightarrow f$
  - empty: $a$
  - infinite: $a \rightarrow e \rightarrow b \rightarrow a \rightarrow e \rightarrow b \rightarrow \cdots$
• $\leftarrow$ inverse of $\rightarrow$

• $\rightarrow^*$ transitive and reflexive closure of $\rightarrow$

• $^* \leftarrow$ inverse of $\rightarrow^*$

$$s \leftrightarrow_R t \iff s \rightarrow_R t \text{ or } t \rightarrow_R s$$

$$s \leftrightarrow^*_R t \iff s = s_0 \leftrightarrow_R s_1 \leftrightarrow_R \ldots \leftrightarrow_R s_n = t \text{ for } n \geq 0$$

• $\leftrightarrow$ symmetric closure of $\rightarrow$

• $\leftrightarrow^*$ conversion (equivalence relation generated by $\rightarrow$) **

• $\rightarrow^+$ transitive closure of $\rightarrow$

• $\rightarrow^=$ reflexive closure of $\rightarrow$

• is relation composition: $R \cdot S = \{ (a, c) \mid a R b \text{ and } b S c \}$
We denote $\to^*$ (resp. $\to^=$) the transitive-reflexive (resp. reflexive) closure of $\to$.

If $\to_1, \to_2$ are binary relations on $A$ then $\to_1 \cdot \to_2$ denotes their composition, i.e. $t \to_1 \cdot \to_2 s$ iff there exists $u \in A$ such that $t \to_1 u \to_2 s$.

We write $(A, \{\to_1, \to_2\})$ to denote the ARS $(A, \to)$ where $\to = \to_1 \cup \to_2$. 
Closure

The transitive-reflexive closure of a relation is a closure operator, i.e. satisfies

\[ \rightarrow \subseteq \rightarrow^*, \quad (\rightarrow^*)^* = \rightarrow^*, \quad \rightarrow_1 \subseteq \rightarrow_2 \text{ implies } \rightarrow_1^* \subseteq \rightarrow_2^* \]

As a consequence

\[ (\rightarrow_1 \cup \rightarrow_2)^* = (\rightarrow_1^* \cup \rightarrow_2^*). \]
### Terminology

- if $x \rightarrow^* y$ then $x$ rewrites to $y$ and $y$ is **reduct** of $x$
- if $x \rightarrow^* z \leftarrow y$ then $z$ is **common reduct** of $x$ and $y$
- if $x \leftrightarrow^* y$ then $x$ and $y$ are **convertible**

### Example

```
\[ \begin{array}{c}
  a & \leftarrow & b & \rightarrow & c & \rightarrow & d \\
  & \uparrow & & \downarrow & & \downarrow \\
  e & \leftarrow & f \\
  & \downarrow & & \downarrow \\
  & & g
\end{array} \]
```

- $a \rightarrow^* f$
- $e \downarrow f$  $f \downarrow d$  not $g \downarrow d$
- $g \leftrightarrow^* d$
Definition 1.1.11. Let $\mathcal{A} = \langle A, \rightarrow \rangle$ be an ARS. An element $a \in A$ is reducible if there exists an element $b \in A$ with $a \rightarrow b$. A normal form is an element that is not reducible. The set of normal forms of $\mathcal{A}$ is denoted by $\text{NF}(\mathcal{A})$ or $\text{NF}(\rightarrow)$ when $A$ can be inferred from the context. An element $a \in A$ has a normal form if $a \rightarrow^* b$ for some normal form $b$. In that case we write $a \rightarrow^! b$.

Element $a$ has normal forms?  
How many normal forms has this ARS?

ARS $\mathcal{A} = \langle A, \rightarrow \rangle$

- $d$ is normal form
- $\text{NF}(\mathcal{A}) = \{ d, g \}$
- $b \rightarrow^! g$
- SN  strong normalization   termination
  - no infinite rewrite sequences

- WN  (weak) normalization
  - every element has at least one normal form
  - $\forall a \exists b \ a \rightarrow^! b$

- UN  unique normal forms
  - no element has more than one normal form
  - $\forall a, b, c \ \text{if } a \rightarrow^! b \ \text{and } a \rightarrow^! c \ \text{then } b = c$
**Termination**

**Definition 1.2.1.** Let $A = \langle A, \to \rangle$ be an ARS. An element $a \in A$ is called *terminating* or *strongly normalizing* (SN) if there are no infinite rewrite sequences starting at $a$. The ARS $A$ is terminating or strongly normalizing if all its elements are terminating. An element $a \in A$ has *unique normal forms* (UN) if it does not have different normal forms ($\forall b, c \in A$ if $a \to^1 b$ and $a \to^1 c$ then $b = c$). The ARS $A$ has unique normal forms if all its elements have unique normal forms.

An element $a$ is *weakly normalizing* (WN) (or simply *normalizing*) if it has a normal form.

\[ a \leftarrow b \rightarrow c \rightarrow d \]

\[ e \leftarrow f \rightarrow g \]

- a is WN? SN?
- c is WN? SN?
- a or c has UN?
- The nf are convertible?
**Confluence**

Definition 1.2.3. Let $\mathcal{A} = \langle A, \rightarrow \rangle$ be an ARS. An element $a \in A$ is confluent if for all elements $b, c \in A$ with $b \overset{*}{\leftarrow} a \overset{*}{\rightarrow} c$ we have $b \downarrow c$. The ARS $\mathcal{A}$ is confluent if all its elements are confluent.

\[ \forall a, b, c \exists d \]

Every confluent ARS has unique normal forms.
1. a is confluent?
2. f is confluent?

3. Can you add a single arrow so that the resulting ARS has **unique normal forms without being confluent**?
Given

\[ R = \begin{cases} 
  f(x, x) & \rightarrow & c \\
  a & \rightarrow & b \\
  f(x, b) & \rightarrow & d
\end{cases} \]

\textcolor{red}{f(a,a)} has normal form?
Can you produce two different \( \text{nf} \)?

we can compute from the same term \( f(a, a) \) two different normal-forms \( c \) and \( d \)
different meaning for equivalent terms
(different meaning for same term!)
Same meaning for *equivalent* terms
Definition 1.2.3. Let $\mathcal{A} = \langle A, \rightarrow \rangle$ be an ARS. An element $a \in A$ is *confluent* if for all elements $b, c \in A$ with $b \leftarrow a \rightarrow^* c$ we have $b \downarrow c$. The ARS $\mathcal{A}$ is confluent if all its elements are confluent.

\[ \text{An ARS } \mathcal{A} = \langle A, \rightarrow \rangle \text{ is confluent if and only if } \leftrightarrow^* \subseteq \downarrow. \]
in an ARS with the property UNC every equivalence class of convertible elements contains at most one normal form.

Q: are UN and UNC equivalent?
Global vs Local
A property of term $t$ is *local* if it is quantified over only *one-step reductions* from $t$; it is *global* if it is quantified over all *rewrite sequences* from $t$.

**Locally confluent (WCR)**  
**Strongly confluent**  
**Diamond**

---

**confluence**  
Let $A = \langle A, \rightarrow \rangle$ be an ARS. An element $a \in A$ is *confluent* if for all elements $b, c \in A$ with $b \overset{*}{\leftarrow} a \overset{*}{\rightarrow} c$ we have $b \downarrow c$. The ARS $A$ is confluent if all its elements are confluent.

**Global property:**
A property of term $t$ is *local* if it is quantified over only *one-step reductions* from $t$; it is *global* if it is quantified over all *rewrite sequences* from $t$.

**Locally confluent (WCR)**

**Strongly confluent**

**Diamond**

Let $\mathcal{A} = \langle A, \rightarrow \rangle$ be an ARS. An element $a \in A$ is *locally confluent* for all elements $b, c \in A$ with $b \rightarrow a \rightarrow c$ we have $b \downarrow c$. The ARS $\mathcal{A}$ is confluent if all its elements are confluent.

An ARS $\mathcal{A} = \langle A, \rightarrow \rangle$ has the *diamond property* ($\Diamond$) if $\leftarrow \cdot \rightarrow \subseteq \rightarrow \cdot \leftarrow$
- diamond property

- $\leftarrow \cdot \rightarrow \subseteq \rightarrow \cdot \leftarrow$

- $\forall a, b, c$

- $\exists d$

- every ARS with diamond property is confluent
An ARS $\mathcal{A} = \langle A, \rightarrow \rangle$ is strongly confluent (SCR) if $\leftrightarrow \cdot \rightarrow \subseteq \rightarrow^\ast \cdot \leftarrow$, see Figure

- Show that every strongly confluent ARS is confluent.
- Does the converse also hold?
- Show that an ARS $\mathcal{A} = \langle A, \rightarrow \rangle$ is confluent if and only if $\leftrightarrow^\ast \cdot \rightarrow \subseteq \rightarrow^\ast \cdot \leftarrow^\ast$. 

---

**Figure:**

- A diagram illustrating the relations between elements in $\mathcal{A}$, with arrows showing the direction of the relations.
Which is true?

1. SN => WN
2. WN => SN

3. Confluence => UN
4. UN => Confluence

5. Confluence => Local confluence
6. Local confluence => Confluence

7. WN & UN => Confluence
8. WN & Local Conf. => Confluence
9. SN & Local Conf. => Confluence
WN vs SN

\[
R = \begin{cases} 
  f(a) \rightarrow c \\
  f(x) \rightarrow f(a)
\end{cases}
\]

The system is weakly normalising but not strongly normalising:

Can you find an infinite reduction sequence?

\[f(b) \rightarrow f(a) \rightarrow c\]

\[f(b) \rightarrow f(a) \rightarrow f(a) \ldots\]
1. SN => WN
2. WN => SN
3. Confluence => UN
4. UN => Confluence
5. Confluence => Local confluence
6. Local confluence => Confluence
7. WN & UN => Confluence
8. WN & Local Conf. => Confluence
9. SN & Local Conf. => Confluence

Newman’s Lemma
Lemma

WN & UN $\implies$ CR

Proof

- WN $\implies \exists n_1, n_2: b_1 \rightarrowleft n_1$ and $b_2 \rightarrowleft n_2$

- UN $\implies n_1 = n_2 \implies b_1 \downarrow b_2$
Newman’s Lemma. *Every terminating and locally confluent ARS is confluent.*

By well-founded induction
Memo: Well-founded Induction

Définition : [Relation bien fondée] Une relation d’ordre $\preceq E \times E$ est bien fondée si il n’existe pas de suite infinie d’éléments de $E$ décroissante par rapport à $>$. 

Theorem : [Principe d’induction bien fondée] Soient donnés un ensemble $E$ quelconque, un ordre strict $<$ sur $E$ (dont $\mathcal{M}$ est son ensemble d’éléments minimaux), et une propriété $P$ sur $E$.

Si

1. pour tout élément minimal $m \in \mathcal{M}$ on a $P(m)$
2. le fait que $P(k)$ soit vérifiée pour tout élément $k < x$ implique $P(x)$

alors

pour tout $x \in E$ on a $P(x)$

The proof technique of well-founded induction states that a property $\mathcal{P}$ of elements of a terminating ARS $\mathcal{A} = \langle A, \rightarrow \rangle$ holds for all elements in $A$ if the following condition is satisfied: An element $a \in A$ has the property $\mathcal{P}$ if all elements $b$ with $a \rightarrow b$ have the property $\mathcal{P}$. In particular every normal form has to satisfy the property $\mathcal{P}$. 
Newman’s Lemma.  *Every terminating and locally confluent ARS is confluent.*

① WCR
② induction hypothesis \((a \rightarrow b_1 \implies b_1 \text{ is CR})\)
③ induction hypothesis \((a \rightarrow c_1 \implies c_1 \text{ is CR})\)
Newman's Lemma. Every terminating and locally confluent ARS is confluent.

A second Proof. Let \( A = \langle A, \rightarrow \rangle \) be terminating and locally confluent.

It suffices to show that every element has unique normal forms.

- Suppose \( B = \{ a \in A \mid \neg \text{UN}(a) \} \neq \emptyset \)
- Let \( b \in B \) be minimal element (with respect to \( \rightarrow \))
- \( b \xrightarrow{!} n_1 \) and \( b \xrightarrow{!} n_2 \) with \( n_1 \neq n_2 \)

\[ \Rightarrow \text{Conclude by showing that it is impossible (absurd)} \]