Rewriting

Part 1. Abstract Reduction

Temur Kutsia

RISC, JKU Linz

Literature

- Franz Baader and Tobias Nipkow. Term Rewriting and All That. Cambridge University Press, 1998.
- Book's home page: http://www21.in.tum.de/~nipkow/TRaAT/
- Resources about rewriting: http://rewriting.loria.fr/

Equational Reasoning

- Restricted class of languages: The only predicate symbol is equality $\approx$.
- Reasoning with equations:
  - derive consequences of given equations,
  - find values for variables that satisfy a given equation.
- At the heart of many problems in mathematics and computer science.

Example: Addition of Natural Numbers

- Equations (identities):

$$\begin{aligned} x + 0 &\approx x \\ x + s(y) &\approx s(x + y) \end{aligned}$$

- How to calculate $s(0) + s(s(0))$?

Example: Addition of Natural Numbers

- Orient equations, obtaining rewriting rules.
- Apply the rules to transform expressions.

Example: Addition of Natural Numbers

- Orient equations, obtaining rewriting rules.
- Apply the rules to transform expressions.
- Rewrite rules:

$$
\begin{aligned}
x + 0 &\to x && (R_1)\\
x + s(y) &\to s(x + y) && (R_2)
\end{aligned}
$$

- Rewriting $s(0) + s(s(0))$:

Example: Addition of Natural Numbers

- Orient equations, obtaining rewriting rules.
- Apply the rules to transform expressions.
- Rewrite rules:

$$
\begin{array}{rcl}
x + 0 \to x & \qquad\qquad & (R_1) \\
x + s(y) \to s(x + y) & \qquad\qquad & (R_2)
\end{array}
$$

- Rewriting $s(0) + s(s(0))$:

$$s(0) + s(s(0)) \to \qquad \text{(by } R_2\text{, with } x \mapsto s(0), y \mapsto s(0)\text{)}$$

Example: Addition of Natural Numbers

- Orient equations, obtaining rewriting rules.
- Apply the rules to transform expressions.
- Rewrite rules:

$$x + 0 \to x \qquad (R_1)$$

$$x + s(y) \to s(x + y) \qquad (R_2)$$

- Rewriting $s(0) + s(s(0))$:

$$s(0) + s(s(0)) \to \qquad \text{(by } R_2\text{, with } x \mapsto s(0), y \mapsto s(0)\text{)}$$

$$s(s(0) + s(0)) \to \qquad \text{(by } R_2\text{, with } x \mapsto s(0), y \mapsto 0\text{)}$$

Example: Addition of Natural Numbers

- Orient equations, obtaining rewriting rules.
- Apply the rules to transform expressions.
- Rewrite rules:

$$
\begin{aligned}
x + 0 &\rightarrow x && (R_1)\\
x + s(y) &\rightarrow s(x + y) && (R_2)
\end{aligned}
$$

- Rewriting $s(0) + s(s(0))$:

$$
\begin{aligned}
s(0) + s(s(0)) &\rightarrow && (\text{by } R_2\text{, with } x \mapsto s(0), y \mapsto s(0))\\
s(s(0) + s(0)) &\rightarrow && (\text{by } R_2\text{, with } x \mapsto s(0), y \mapsto 0)\\
s(s(s(0) + 0)) &\rightarrow && (\text{by } R_1\text{, with } x \mapsto s(0))
\end{aligned}
$$

Example: Addition of Natural Numbers

- Orient equations, obtaining rewriting rules.
- Apply the rules to transform expressions.
- Rewrite rules:

$$
\begin{aligned}
x + 0 &\to x && (R_1) \\
x + s(y) &\to s(x + y) && (R_2)
\end{aligned}
$$

- Rewriting $s(0) + s(s(0))$:

$$
\begin{aligned}
s(0) + s(s(0)) &\to && \text{(by } R_2\text{, with } x \mapsto s(0), y \mapsto s(0)\text{)} \\
s(s(0) + s(0)) &\to && \text{(by } R_2\text{, with } x \mapsto s(0), y \mapsto 0\text{)} \\
s(s(s(0) + 0)) &\to && \text{(by } R_1\text{, with } x \mapsto s(0)\text{)} \\
s(s(s(0))) & &&
\end{aligned}
$$

What is Rewriting

- Process of transforming one expression into another.
- Rules describe how one expression can be rewritten into another.

Identities and Rewriting

- Rewriting as a computational mechanism:
    - Apply given equations in one direction, as rewrite rules.
    - Compute normal forms.
    - Close relationship with functional programming.
    - Example: symbolic differentiation.

Identities and Rewriting

- Rewriting as a computational mechanism:
    - Apply given equations in one direction, as rewrite rules.
    - Compute normal forms.
    - Close relationship with functional programming.
    - Example: symbolic differentiation.
- Rewriting as a deduction mechanism:
    - Apply given equations in both directions.
    - Define equivalence classes of terms.
    - Equational reasoning.
    - Example: group theory.

Symbolic Differentiation

- Expressions: Terms built over variables $(u, v, \ldots)$ and the following function symbols:
    - constants $0, 1$ (numbers),
    - constants X, Y (indeterminates),
    - unary symbol D_X (partial derivative with respect to X),
    - binary symbols $+, *$.

Symbolic Differentiation

- Expressions: Terms built over variables $(u, v, \dots)$ and the following function symbols:
  - constants $0, 1$ (numbers),
  - constants X, Y (indeterminates),
  - unary symbol D_X (partial derivative with respect to X),
  - binary symbols $+, *$.
- Examples of terms:
  - $(X + X) * Y + 1$.
  - $D_X(u * v)$.
  - $(X + Y) * D_X(X * Y)$.

Symbolic Differentiation

- Expressions: Terms built over variables $(u, v, \ldots)$ and the following function symbols:
  - constants $0, 1$ (numbers),
  - constants X, Y (indeterminates),
  - unary symbol D_X (partial derivative with respect to X),
  - binary symbols $+, *$.
- Examples of terms:
  - $(X + X) * Y + 1$.
  - $D_X(u * v)$.
  - $(X + Y) * D_X(X * Y)$.
- Rewrite rules:

$$\begin{aligned} D_X(X) &\to 1 && (R_1) \\ D_X(Y) &\to 0 && (R_2) \\ D_X(u + v) &\to D_X(u) + D_X(v) && (R_3) \\ D_X(u * v) &\to (u * D_X(v)) + (D_X(u) * v) && (R_4) \end{aligned}$$

Symbolic Differentiation

- Rewrite rules:

$$D_X(X) \to 1 \tag{R_1}$$

$$D_X(Y) \to 0 \tag{R_2}$$

$$D_X(u + v) \to D_X(u) + D_X(v) \tag{R_3}$$

$$D_X(u * v) \to (u * D_X(v)) + (D_X(u) * v) \tag{R_4}$$

- Differentiate $D_X(X * X)$:

$$D_X(X * X)$$

Symbolic Differentiation

- Rewrite rules:

$$D_X(X) \to 1 \tag{R_1}$$

$$D_X(Y) \to 0 \tag{R_2}$$

$$D_X(u + v) \to D_X(u) + D_X(v) \tag{R_3}$$

$$D_X(u * v) \to (u * D_X(v)) + (D_X(u) * v) \tag{R_4}$$

- Differentiate $D_X(X * X)$:

$$\begin{tikzcd}
D_X(X * X) \arrow[d, "(R_4)"] \\
(X * D_X(X)) + (D_X(X) * X)
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$
\begin{aligned}
D_X(X) &\to 1 \qquad (R_1)\\
D_X(Y) &\to 0 \qquad (R_2)\\
D_X(u+v) &\to D_X(u) + D_X(v) \qquad (R_3)\\
D_X(u*v) &\to (u*D_X(v)) + (D_X(u)*v) \qquad (R_4)
\end{aligned}
$$

- Differentiate $D_X(X*X)$:

$$
\begin{array}{c}
D_X(X*X)\\
\Big\downarrow {\scriptstyle (R_4)}\\
(X*D_X(X)) + (D_X(X)*X)
\end{array}
$$

Symbolic Differentiation

- Rewrite rules:

$$
\begin{aligned}
D_X(X) &\to 1 && (R_1)\\
D_X(Y) &\to 0 && (R_2)\\
D_X(u+v) &\to D_X(u) + D_X(v) && (R_3)\\
D_X(u*v) &\to (u * D_X(v)) + (D_X(u) * v) && (R_4)
\end{aligned}
$$

- Differentiate $D_X(X * X)$:

$$\begin{tikzcd}
 & D_X(X * X) \arrow[d, "(R_4)"] \\
 & (X * D_X(X)) + (D_X(X) * X) \arrow[dl, "(R_1)"'] \\
(X * 1) + (D_X(X) * X) &
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$D_X(X) \to 1 \tag{R_1}$$

$$D_X(Y) \to 0 \tag{R_2}$$

$$D_X(u + v) \to D_X(u) + D_X(v) \tag{R_3}$$

$$D_X(u * v) \to (u * D_X(v)) + (D_X(u) * v) \tag{R_4}$$

- Differentiate $D_X(X * X)$:

$$\begin{tikzcd}
 & D_X(X * X) \arrow[d, "(R_4)"] \\
 & (X * D_X(X)) + (D_X(X) * X) \arrow[dl, "(R_1)"'] \\
(X * 1) + (D_X(X) * X) &
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$
\begin{aligned}
D_X(X) &\to 1 && (R_1)\\
D_X(Y) &\to 0 && (R_2)\\
D_X(u+v) &\to D_X(u)+D_X(v) && (R_3)\\
D_X(u*v) &\to (u*D_X(v))+(D_X(u)*v) && (R_4)
\end{aligned}
$$

- Differentiate $D_X(X*X)$:

$$\begin{tikzcd}
 & D_X(X*X) \arrow[d, "(R_4)"] \\
 & (X*D_X(X))+(D_X(X)*X) \arrow[dl, "(R_1)"'] \\
(X*1)+(D_X(X)*X) &
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$D_X(X) \to 1 \tag{R_1}$$

$$D_X(Y) \to 0 \tag{R_2}$$

$$D_X(u + v) \to D_X(u) + D_X(v) \tag{R_3}$$

$$D_X(u * v) \to (u * D_X(v)) + (D_X(u) * v) \tag{R_4}$$

- Differentiate $D_X(X * X)$:

$$\begin{tikzcd}
& D_X(X * X) \arrow[d, "(R_4)"] & \\
& (X * D_X(X)) + (D_X(X) * X) \arrow[dl, "(R_1)"'] & \\
(X * 1) + (D_X(X) * X) \arrow[dr, "(R_1)"'] & & \\
& (X * 1) + (1 * X) &
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$\begin{aligned}
D_X(X) &\to 1 && (R_1)\\
D_X(Y) &\to 0 && (R_2)\\
D_X(u+v) &\to D_X(u) + D_X(v) && (R_3)\\
D_X(u*v) &\to (u*D_X(v)) + (D_X(u)*v) && (R_4)
\end{aligned}$$

- Differentiate $D_X(X*X)$:

$$\begin{tikzcd}
 & D_X(X*X) \arrow[d, "(R_4)"] \\
 & (X*D_X(X)) + (D_X(X)*X) \arrow[dl, "(R_1)"'] \\
(X*1) + (D_X(X)*X) \arrow[dr, "(R_1)"'] & \\
 & (X*1) + (1*X)
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$\begin{aligned}
D_X(X) &\to 1 \tag{R_1}\\
D_X(Y) &\to 0 \tag{R_2}\\
D_X(u+v) &\to D_X(u) + D_X(v) \tag{R_3}\\
D_X(u*v) &\to (u*D_X(v)) + (D_X(u)*v) \tag{R_4}
\end{aligned}$$

- Differentiate $D_X(X*X)$:

$$\begin{tikzcd}
 & D_X(X*X) \arrow[d, "(R_4)"] \\
 & (X*D_X(X))) + (D_X(X)*X) \arrow[dl, "(R_1)"'] \\
(X*1) + (D_X(X)*X) \arrow[dr, "(R_1)"'] & \\
 & (X*1) + (1*X)
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$\begin{aligned} D_X(X) &\to 1 && (R_1)\\ D_X(Y) &\to 0 && (R_2)\\ D_X(u+v) &\to D_X(u)+D_X(v) && (R_3)\\ D_X(u*v) &\to (u*D_X(v))+(D_X(u)*v) && (R_4) \end{aligned}$$

- Differentiate $D_X(X*X)$:

$$\begin{tikzcd}
 & D_X(X*X) \arrow[d, "(R_4)"] & \\
 & (X*D_X(X))+(D_X(X)*X) \arrow[dl, "(R_1)"'] \arrow[dr, "(R_1)"] & \\
(X*1)+(D_X(X)*X) \arrow[dr, "(R_1)"'] & & (X*D_X(X))+(1*X) \\
 & (X*1)+(1*X) &
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$
\begin{aligned}
D_X(X) &\to 1 \qquad &(R_1)\\
D_X(Y) &\to 0 \qquad &(R_2)\\
D_X(u+v) &\to D_X(u) + D_X(v) \qquad &(R_3)\\
D_X(u*v) &\to (u * D_X(v)) + (D_X(u) * v) \qquad &(R_4)
\end{aligned}
$$

- Differentiate $D_X(X * X)$:

$$\begin{tikzcd}
 & D_X(X * X) \arrow[d, "(R_4)"] & \\
 & (X * D_X(X)) + (D_X(X) * X) \arrow[dl, "(R_1)"'] \arrow[dr, "(R_1)"] & \\
(X * 1) + (D_X(X) * X) \arrow[dr, "(R_1)"'] & & (X * D_X(X)) + (1 * X) \\
 & (X * 1) + (1 * X) &
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$\begin{align} D_X(X) &\to 1 \tag{R_1} \\ D_X(Y) &\to 0 \tag{R_2} \\ D_X(u+v) &\to D_X(u) + D_X(v) \tag{R_3} \\ D_X(u*v) &\to (u * D_X(v)) + (D_X(u) * v) \tag{R_4} \end{align}$$

- Differentiate $D_X(X*X)$:

$$\begin{tikzcd}
 & D_X(X*X) \arrow[d, "(R_4)"] & \\
 & (X*D_X(X)) + (D_X(X)*X) \arrow[dl, "(R_1)"'] \arrow[dr, "(R_1)"] & \\
(X*1) + (D_X(X)*X) \arrow[dr, "(R_1)"'] & & (X*D_X(X)) + (1*X) \\
 & (X*1) + (1*X) &
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$
\begin{aligned}
D_X(X) &\to 1 && (R_1)\\
D_X(Y) &\to 0 && (R_2)\\
D_X(u+v) &\to D_X(u) + D_X(v) && (R_3)\\
D_X(u*v) &\to (u * D_X(v)) + (D_X(u) * v) && (R_4)
\end{aligned}
$$

- Differentiate $D_X(X * X)$:

$$\begin{tikzcd}
& D_X(X * X) \arrow[d, "(R_4)"] & \\
& (X * D_X(X)) + (D_X(X) * X) \arrow[dl, "(R_1)"'] \arrow[dr, "(R_1)"] & \\
(X * 1) + (D_X(X) * X) \arrow[dr, "(R_1)"'] & & (X * D_X(X)) + (1 * X) \arrow[dl, "(R_1)"] \\
& (X * 1) + (1 * X) &
\end{tikzcd}$$

Symbolic Differentiation

- Rewrite rules:

$$\begin{aligned} D_X(X) &\rightarrow 1 \tag{R_1} \\ D_X(Y) &\rightarrow 0 \tag{R_2} \\ D_X(u+v) &\rightarrow D_X(u) + D_X(v) \tag{R_3} \\ D_X(u*v) &\rightarrow (u * D_X(v)) + (D_X(u) * v) \tag{R_4} \end{aligned}$$

- Differentiate $D_X(X * X)$:

$$\begin{tikzcd}
 & D_X(X * X) \arrow[d, "(R_4)"] & \\
 & (X * D_X(X)) + (D_X(X) * X) \arrow[dl, "(R_1)"'] \arrow[dr, "(R_1)"] & \\
(X * 1) + (D_X(X) * X) \arrow[dr, "(R_1)"'] & & (X * D_X(X)) + (1 * X) \arrow[dl, "(R_1)"] \\
 & (X * 1) + (1 * X) &
\end{tikzcd}$$

Properties of Term Rewriting Systems

The symbolic differentiation example can be used to illustrate two most important properties of TRSs:

Properties of Term Rewriting Systems

The symbolic differentiation example can be used to illustrate two most important properties of TRSs:

1. Termination:
   - Is it always the case that after finitely many rule applications we reach an expression to which no more rules apply (normal form)?
   - For symbolic differentiation rules this is the case.
   - But how to prove it?
   - An example of non-terminating rule: $u + v \to v + u$

Properties of Term Rewriting Systems

The symbolic differentiation example can be used to illustrate two most important properties of TRSs:

Properties of Term Rewriting Systems

The symbolic differentiation example can be used to illustrate two most important properties of TRSs:

2. Confluence:
   - If there are different ways of applying rules to a given term t, leading to different terms t_1 and t_2, can they be reduced by rule applications to a common term?
   - For symbolic differentiation rules this is the case.
   - But how to prove it?

Properties of Term Rewriting Systems

- Adding the rule $u + 0 \to u$ (R_5) destroys confluence:

$$\begin{tikzcd}
& D_X(X+0) \arrow[dl, "(R_5)"'] \arrow[dr, "(R_3)"] & \\
D_X(X) \arrow[d, "(R_1)"'] & & D_X(X) + D_X(0) \arrow[d, "(R_1)"] \\
1 & & 1 + D_X(0)
\end{tikzcd}$$

- Confluence can be regained by adding $D_X(0) \to 0$ (completion).

Group Theory

- Terms are built over variables and the following function symbols:
    - binary $\circ$,
    - unary i,
    - constant 0.
- Examples of terms:
    - $x \circ (y \circ i(y))$
    - $(0 \circ x) \circ i(0)$
    - $i(x \circ y)$
- Identities (aka group axioms), defining groups:

Associativity of $\circ$	$(x \circ y) \circ z \approx x \circ (y \circ z)$	(G_1)
e left unit	$e \circ x \approx x$	(G_2)
i left inverse	$i(x) \circ x \approx e$	(G_3)

Group Theory

- Identities can be applied in both directions.
- Word problem for identities:
  - Given a set of identities E and two terms s and t.
  - Is it possible to transform s into t, using the identities in E as rewrite rules applied in both directions?
- For instance, is it possible to transform e into $x \circ i(x)$, i.e., is the left inverse also a right-inverse?

Group Theory

$$
\begin{aligned}
(x \circ y) \circ z &\approx x \circ (y \circ z) && (G_1)\\
e \circ x &\approx x && (G_2)\\
i(x) \circ x &\approx e && (G_3)
\end{aligned}
$$

Transform e into $x \circ i(x)$:

$$
\begin{aligned}
e &\approx_{G_3} i(x \circ i(x)) \circ (x \circ i(x))\\
&\approx_{G_2} i(x \circ i(x)) \circ (x \circ (e \circ i(x)))\\
&\approx_{G_3} i(x \circ i(x)) \circ (x \circ ((i(x) \circ x) \circ i(x)))\\
&\approx_{G_1} i(x \circ i(x)) \circ ((x \circ (i(x) \circ x)) \circ i(x))\\
&\approx_{G_1} i(x \circ i(x)) \circ (((x \circ i(x)) \circ x) \circ i(x))\\
&\approx_{G_1} i(x \circ i(x)) \circ ((x \circ i(x)) \circ (x \circ i(x)))\\
&\approx_{G_1} (i(x \circ i(x)) \circ (x \circ i(x))) \circ (x \circ i(x))\\
&\approx_{G_3} e \circ (x \circ i(x))\\
&\approx_{G_3} x \circ i(x)
\end{aligned}
$$

Solving Word Problems by Rewriting?

- Is there a simpler way to solve word problems?
- Try to solve it by rewriting (uni-directional application of identities):

$$\begin{tikzcd}
s \arrow[dr, "*"'] & & & & t \arrow[dl, "*"] \\
& \hat{s} & = & \hat{t} &
\end{tikzcd}$$

- Reduce s and t to normal forms $\hat{s}$ and $\hat{t}$.
- Check whether $\hat{s} = \hat{t}$, i.e., syntactically equal.
  (= is the meta-equality.)

Solving Word Problems by Rewriting?

- Is there a simpler way to solve word problems?
- Try to solve it by rewriting (uni-directional application of identities):

$$\begin{tikzcd}
s \arrow[dr, "*"'] & & & t \arrow[dl, "*"] \\
& \hat{s} \arrow[r, phantom, "="] & \hat{t} &
\end{tikzcd}$$

- Reduce s and t to normal forms $\hat{s}$ and $\hat{t}$.
- Check whether $\hat{s} = \hat{t}$, i.e., syntactically equal.
  (= is the meta-equality.)
- But... it would only work if normal forms exist and are unique.

Solving Word Problems by Rewriting?

- In the group theory example, e and $x \circ i(x)$ are equivalent, but it can not be decided by (left-to-right) rewriting: Both terms are in the normal form.
- Uniqueness of normal forms is violated: non-confluence.
- Normal forms may not exist: The process of reducing a term may lead to an infinite chain of transformations: non-termination.

Solving Word Problems by Rewriting?

- In the group theory example, e and $x \circ i(x)$ are equivalent, but it can not be decided by (left-to-right) rewriting: Both terms are in the normal form.
- Uniqueness of normal forms is violated: non-confluence.
- Normal forms may not exist: The process of reducing a term may lead to an infinite chain of transformations: non-termination.
- Termination and confluence ensure existence and uniqueness of normal forms.

Solving Word Problems by Rewriting?

- In the group theory example, e and $x \circ i(x)$ are equivalent, but it can not be decided by (left-to-right) rewriting: Both terms are in the normal form.
- Uniqueness of normal forms is violated: non-confluence.
- Normal forms may not exist: The process of reducing a term may lead to an infinite chain of transformations: non-termination.
- Termination and confluence ensure existence and uniqueness of normal forms.
- If a given set of identities leads to non-confluent system, we will try to apply the idea of completion to extend the rewrite system to a confluent one.

Abstract vs Concrete

Concrete rewrite formalisms:

- string rewriting
- term rewriting
- graph rewriting
- λ calculus
- etc.

Abstract reduction:

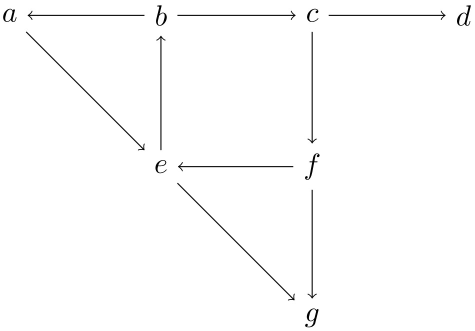
- No structure on objects to be rewritten.
- Abstract treatment of reductions.

Abstract Reduction Systems

- Abstract reduction system (ARS): A pair $(A, \to)$, where
  - A is a set,
  - the reduction $\to$ is a binary relation on A: $\to \subseteq A \times A$.
- Write $a \to b$ for $(a, b) \in \to$.

Abstract Reduction System: Example

- $A = \{a, b, c, d, e, f, g\}$
- $\rightarrow = \left\{ \begin{array}{l} (a,e),(b,a),(b,c),(c,d),(c,f) \\ (e,b),(e,g),(f,e),(f,g) \end{array} \right\}$



Equivalence and Reduction

Again, two views at reductions.

1. Directed computation: Follow the reductions, trying to compute a normal form: $a_0 \to a_1 \to \cdots$
2. View $\to$ as description of $\overset{*}{\leftrightarrow}$.
    - $a \overset{*}{\leftrightarrow} b$ means there is a path between a and b, with arrows traversed in both directions: $a \leftarrow c \to d \leftarrow b$
    - Goal: Decide whether $a \overset{*}{\leftrightarrow} b$.
    - Bidirectional rewriting is expensive.
    - Unidirectional rewriting with subsequent comparison of normal form works if the reduction system is confluent and terminating.

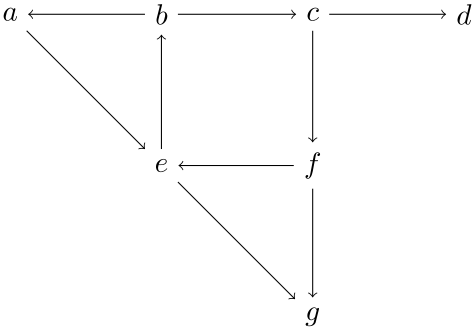
Termination, confluence: central topics.

Basic notions

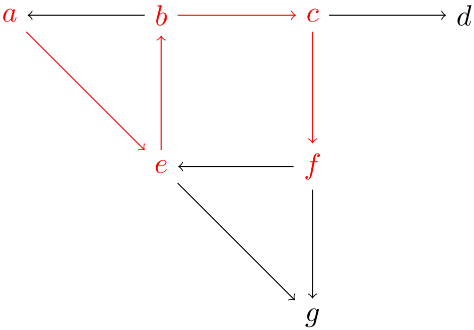
1. Composition of two relations.
2. Given two relations $R \subseteq A \times B$ and $S \subseteq B \times C$, their composition is defined by

$$R \circ S := \{(x, z) \mid \exists y \in B.\ (x, y) \in R \wedge (y, z) \in S\}$$

Abstract Reduction System: Example

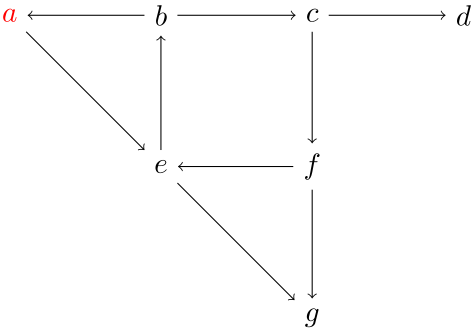


Abstract Reduction System: Example



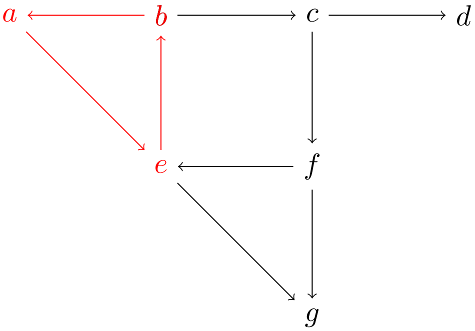
- Finite rewrite sequence: $a \to e \to b \to c \to f$

Abstract Reduction System: Example



- Finite rewrite sequence: $a \to e \to b \to c \to f$
- Empty rewrite sequence: a

Abstract Reduction System: Example



- Finite rewrite sequence: $a \to e \to b \to c \to f$
- Empty rewrite sequence: a
- Infinite rewrite sequence: $a \to e \to b \to a \to \cdots$

Relations Derived from →

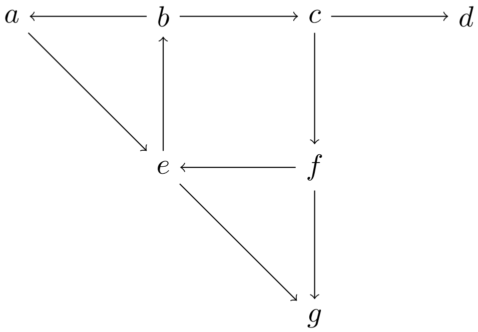
$\xrightarrow{0} := \{(x,x) \mid x \in A\}$	identity
$\xrightarrow{i+1} := \xrightarrow{i} \circ \rightarrow$	$(i+1)$-fold composition, $i \geq 0$
$\xrightarrow{+} := \cup_{i>0} \xrightarrow{i}$	transitive closure
$\xrightarrow{*} := \xrightarrow{+} \cup \xrightarrow{0}$	reflexive transitive closure
$\xrightarrow{=} := \rightarrow \cup \xrightarrow{0}$	reflexive closure
$\xrightarrow{-1} := \{(y,x) \mid (x,y) \in \rightarrow\}$	inverse
$\leftarrow := \xrightarrow{-1}$	inverse
$\leftrightarrow := \rightarrow \cup \leftarrow$	symmetric closure
$\overset{+}{\leftrightarrow} := (\leftrightarrow)^{+}$	transitive symmetric closure
$\overset{*}{\leftrightarrow} := (\leftrightarrow)^{*}$	reflexive transitive symmetric closure

Terminology

- If $x \xrightarrow{*} y$ then we say:
  - x rewrites to y, or
  - there is some finite path from x to y, or
  - y is a reduct of x.

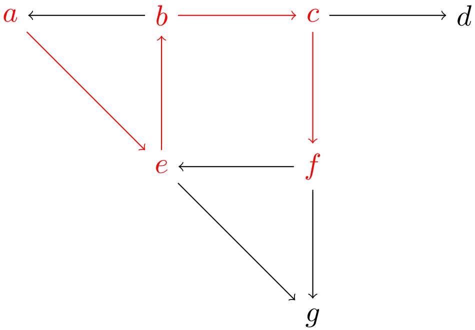
Terminology

- If $x \xrightarrow{*} y$ then we say:
  - x rewrites to y, or
  - there is some finite path from x to y, or
  - y is a reduct of x.



Terminology

- If $x \xrightarrow{*} y$ then we say:
    - x rewrites to y, or
    - there is some finite path from x to y, or
    - y is a reduct of x.



$$a \xrightarrow{*} f$$

Terminology

- x is reducible iff there exists y such that $x \to y$.

Terminology

- x is reducible iff there exists y such that $x \to y$.
- x is in normal form (irreducible) iff x is not reducible.

Terminology

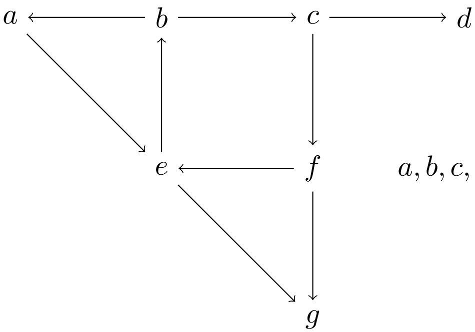
- x is reducible iff there exists y such that $x \to y$.
- x is in normal form (irreducible) iff x is not reducible.
- y is a normal form of x iff $x \xrightarrow{*} y$ and y is in normal form.

Terminology

- x is reducible iff there exists y such that $x \to y$.
- x is in normal form (irreducible) iff x is not reducible.
- y is a normal form of x iff $x \xrightarrow{*} y$ and y is in normal form.
- We write $x \xrightarrow{!} y$ if y is a normal form of x.
- If x has a unique normal form, it is denoted by $x\downarrow$.

Terminology

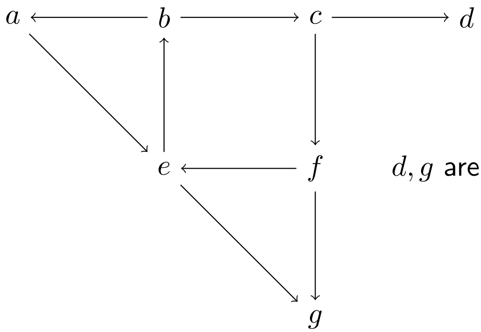
- x is reducible iff there exists y such that $x \to y$.
- x is in normal form (irreducible) iff x is not reducible.
- y is a normal form of x iff $x \xrightarrow{*} y$ and y is in normal form.
- We write $x \xrightarrow{!} y$ if y is a normal form of x.
- If x has a unique normal form, it is denoted by $x\downarrow$.



a, b, c, e, f are reducible

Terminology

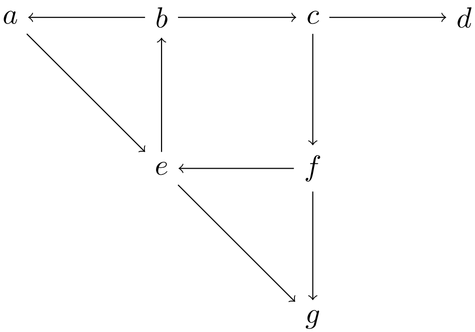
- x is reducible iff there exists y such that $x \to y$.
- x is in normal form (irreducible) iff x is not reducible.
- y is a normal form of x iff $x \xrightarrow{*} y$ and y is in normal form.
- We write $x \xrightarrow{!} y$ if y is a normal form of x.
- If x has a unique normal form, it is denoted by $x\downarrow$.



d, g are in a normal form

Terminology

- x is reducible iff there exists y such that $x \to y$.
- x is in normal form (irreducible) iff x is not reducible.
- y is a normal form of x iff $x \xrightarrow{*} y$ and y is in normal form.
- We write $x \xrightarrow{!} y$ if y is a normal form of x.
- If x has a unique normal form, it is denoted by $x\downarrow$.



$$b \xrightarrow{!} d$$
$$b \xrightarrow{!} g$$
$$g \xrightarrow{!} g$$

Terminology

- y is direct successor of x iff $x \to y$.

Terminology

- y is direct successor of x iff $x \to y$.
- y is successor of x iff $x \xrightarrow{+} y$.

Terminology

- y is direct successor of x iff $x \to y$.
- y is successor of x iff $x \xrightarrow{+} y$.
- x and y are convertible iff $x \overset{*}{\leftrightarrow} y$.

Terminology

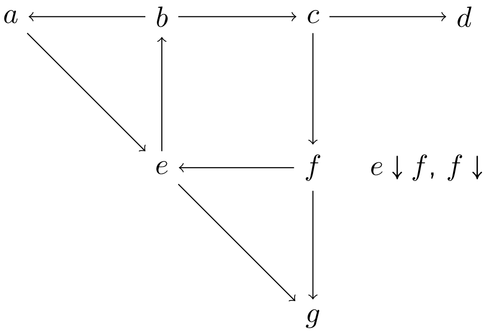
- y is direct successor of x iff $x \to y$.
- y is successor of x iff $x \xrightarrow{+} y$.
- x and y are convertible iff $x \overset{*}{\leftrightarrow} y$.
- x and y are joinable iff there exists z such that $x \xrightarrow{*} z \xleftarrow{*} y$.

Terminology

- y is direct successor of x iff $x \to y$.
- y is successor of x iff $x \xrightarrow{+} y$.
- x and y are convertible iff $x \overset{*}{\leftrightarrow} y$.
- x and y are joinable iff there exists z such that $x \xrightarrow{*} z \xleftarrow{*} y$.
- We write $x \downarrow y$ iff x and y are joinable.

Terminology

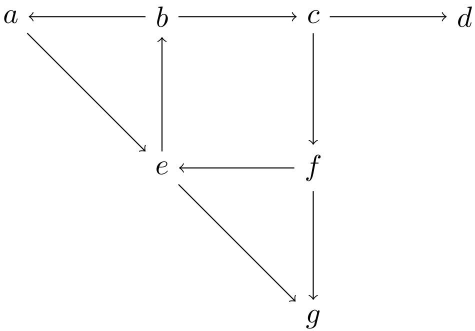
- y is direct successor of x iff $x \rightarrow y$.
- y is successor of x iff $x \xrightarrow{+} y$.
- x and y are convertible iff $x \overset{*}{\leftrightarrow} y$.
- x and y are joinable iff there exists z such that $x \xrightarrow{*} z \xleftarrow{*} y$.
- We write $x \downarrow y$ iff x and y are joinable.



$e \downarrow f$, $f \downarrow d$, $a \downarrow f$, not $g \downarrow d$

Terminology

- y is direct successor of x iff $x \to y$.
- y is successor of x iff $x \xrightarrow{+} y$.
- x and y are convertible iff $x \overset{*}{\leftrightarrow} y$.
- x and y are joinable iff there exists z such that $x \xrightarrow{*} z \xleftarrow{*} y$.
- We write $x \downarrow y$ iff x and y are joinable.



$g \overset{*}{\leftrightarrow} d$

Central Notions

Definition 1.1

A reduction → is called Church-Rosser (CR) iff

$$x \overset{*}{\leftrightarrow} y \text{ implies } x \downarrow y.$$

Central Notions

Definition 1.1

A reduction → is called Church-Rosser (CR) iff

$$x \overset{*}{\leftrightarrow} y \text{ implies } x \downarrow y.$$

Graphically:

$$\begin{tikzcd}
x \arrow[rr, leftrightarrow, "*"] \arrow[dr, dashed, "*"'] & & y \arrow[dl, dashed, "*"] \\
& z &
\end{tikzcd}$$

Solid arrows represent universal and dashed arrows existential quantification: $\forall x, y.\ x \overset{*}{\leftrightarrow} y \Rightarrow \exists z.\ x \overset{*}{\rightarrow} z \wedge y \overset{*}{\rightarrow} z.$

Central Notions

Definition 1.2

A reduction → is called confluent (C) iff

$$y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2 \text{ implies } y_1 \downarrow y_2.$$

Central Notions

Definition 1.2

A reduction → is called confluent (C) iff

$$y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2 \text{ implies } y_1 \downarrow y_2.$$

Graphically:

$$\begin{tikzcd}
x \arrow[r, "*"] \arrow[d, "*"'] & y_1 \arrow[d, dashed, "*"] \\
y_2 \arrow[r, dashed, "*"'] & z
\end{tikzcd}$$

Solid arrows represent universal and dashed arrows existential quantification: $\forall x, y_1, y_2.\ y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2 \Rightarrow \exists z.\ y_1 \overset{*}{\rightarrow} z \overset{*}{\leftarrow} y_2.$

Central Notions

Definition 1.3

A reduction → is called locally confluent (LC) iff

$$y_1 \leftarrow x \rightarrow y_2 \text{ implies } y_1 \downarrow y_2.$$

Central Notions

Definition 1.3

A reduction → is called locally confluent (LC) iff

$$y_1 \leftarrow x \rightarrow y_2 \text{ implies } y_1 \downarrow y_2.$$

Graphically:

$$\begin{tikzcd}
x \arrow[r] \arrow[d] & y_1 \arrow[d, dashed, "*"] \\
y_2 \arrow[r, dashed, "*"'] & z
\end{tikzcd}$$

Solid arrows represent universal and dashed arrows existential quantification: $\forall x, y_1, y_2.\ y_1 \leftarrow x \rightarrow y_2 \Rightarrow \exists z.\ y_1 \overset{*}{\rightarrow} z \overset{*}{\leftarrow} y_2$.

Central Notions

Definition 1.4

A reduction → is called

- terminating (T) iff there is no infinite descending chain $a_0 \to a_1 \to \cdots$.
- normalizing (N) iff every element has a normal form.
- uniquely normalizing (UN) iff every element has at most one normal form.
- convergent iff it is both confluent and terminating.

Central Notions

Definition 1.4

A reduction → is called

- terminating (T) iff there is no infinite descending chain $a_0 \to a_1 \to \cdots$.
- normalizing (N) iff every element has a normal form.
- uniquely normalizing (UN) iff every element has at most one normal form.
- convergent iff it is both confluent and terminating.

Alternative terminology:

- Strongly normalizing: terminating.
- Weakly normalizing: normalizing.

Central Notions

- Obviously, $x \downarrow y$ implies $x \overset{*}{\leftrightarrow} y$.
- Therefore, the Church-Rosser property can be formulated as the equivalence:
- $\rightarrow$ is called Church-Rosser iff

$$x \overset{*}{\leftrightarrow} y \text{ iff } x \downarrow y.$$

Properties

1. T $\Longrightarrow$ N

Properties

1. $\mathsf{T} \Longrightarrow \mathsf{N}$
2. $\mathsf{T} \not\Longleftarrow \mathsf{N}$

Properties

1. T $\Longrightarrow$ N
2. T $\not\Longleftarrow$ N

$$\begin{tikzcd}a \arrow[loop left] \arrow[r] & b\end{tikzcd}$$

Properties

1. T $\Longrightarrow$ N
2. T $\not\Longleftarrow$ N
3. CR $\Longleftrightarrow$ $\overset{*}{\leftrightarrow} = \downarrow$

$$\begin{tikzcd} a \arrow[loop left] \arrow[r] & b \end{tikzcd}$$

Properties

1. T $\Longrightarrow$ N
2. T $\not\Longleftarrow$ N $\qquad \circlearrowleft a \longrightarrow b$
3. CR $\Longleftrightarrow$ $\overset{*}{\leftrightarrow} = \downarrow$
4. CR $\Longrightarrow$ UN

Properties

1. T $\Longrightarrow$ N
2. T $\not\Longleftarrow$ N $\qquad \circlearrowright a \longrightarrow b$
3. CR $\Longleftrightarrow$ $\overset{*}{\leftrightarrow} = \downarrow$
4. CR $\Longrightarrow$ UN
5. CR $\not\Longleftarrow$ UN

Properties

1. T $\Longrightarrow$ N
2. T $\not\Longleftarrow$ N $\qquad \circlearrowright a \longrightarrow b$
3. CR $\Longleftrightarrow$ $\overset{*}{\leftrightarrow} = \downarrow$
4. CR $\Longrightarrow$ UN
5. CR $\not\Longleftarrow$ UN $\qquad \circlearrowright a \longleftarrow b \longrightarrow c$

Properties

1. $\mathsf{T} \Longrightarrow \mathsf{N}$
2. $\mathsf{T} \not\Longleftarrow \mathsf{N}$ $\qquad \circlearrowleft a \longrightarrow b$
3. $\mathsf{CR} \Longleftrightarrow \overset{*}{\leftrightarrow} = \downarrow$
4. $\mathsf{CR} \Longrightarrow \mathsf{UN}$
5. $\mathsf{CR} \not\Longleftarrow \mathsf{UN}$ $\qquad \circlearrowleft a \longleftarrow b \longrightarrow c$
6. $\mathsf{N} \wedge \mathsf{UN} \Longrightarrow \mathsf{C}$

Properties

1. $\mathsf{T} \Longrightarrow \mathsf{N}$
2. $\mathsf{T} \not\Longleftarrow \mathsf{N}$ $\qquad \circlearrowright a \longrightarrow b$
3. $\mathsf{CR} \Longleftrightarrow \overset{*}{\leftrightarrow} = \downarrow$
4. $\mathsf{CR} \Longrightarrow \mathsf{UN}$
5. $\mathsf{CR} \not\Longleftarrow \mathsf{UN}$ $\qquad \circlearrowright a \longleftarrow b \longrightarrow c$
6. $\mathsf{N} \wedge \mathsf{UN} \Longrightarrow \mathsf{C}$
7. $\mathsf{C} \Longrightarrow \mathsf{LC}$

Properties

1. T $\Longrightarrow$ N
2. T $\not\Longleftarrow$ N $\qquad \circlearrowright\, a \longrightarrow b$
3. CR $\Longleftrightarrow$ $\overset{*}{\longleftrightarrow} = \downarrow$
4. CR $\Longrightarrow$ UN
5. CR $\not\Longleftarrow$ UN $\qquad \circlearrowright\, a \longleftarrow b \longrightarrow c$
6. N $\wedge$ UN $\Longrightarrow$ C
7. C $\Longrightarrow$ LC
8. C $\not\Longleftarrow$ LC

Properties

1. T $\Longrightarrow$ N
2. T $\not\Longleftarrow$ N $\qquad \circlearrowright a \longrightarrow b$
3. CR $\Longleftrightarrow$ $\overset{*}{\leftrightarrow} = \downarrow$
4. CR $\Longrightarrow$ UN
5. CR $\not\Longleftarrow$ UN $\qquad \circlearrowright a \longleftarrow b \longrightarrow c$
6. N $\wedge$ UN $\Longrightarrow$ C
7. C $\Longrightarrow$ LC
8. C $\not\Longleftarrow$ LC

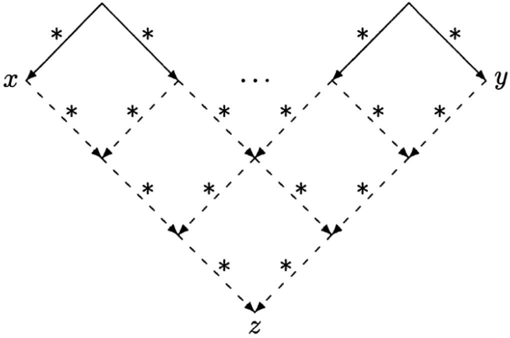
$$\begin{tikzcd} a & b \arrow[l] \arrow[r, bend left] & c \arrow[l, bend left] \arrow[r] & d \end{tikzcd}$$

Properties

- Recall we were looking for:
- Ability to check equivalence by the search of a common reduct.
- This is exactly the Church-Rosser property.
- How does it relate to confluence and termination?

Church-Rosser and Confluence

- The Church-Rosser property and confluence coincide.
- CR $\Longrightarrow$ C is immediate.
- CR $\Longleftarrow$ C has a nice diagrammatic proof:



Church-Rosser and Confluence

Definition 1.5

A reduction → is called semi-confluent (SC) iff

$$y_1 \leftarrow x \overset{*}{\rightarrow} y_2 \text{ implies } y_1 \downarrow y_2.$$

Church-Rosser and Confluence

Definition 1.5

A reduction → is called semi-confluent (SC) iff

$$y_1 \leftarrow x \xrightarrow{*} y_2 \text{ implies } y_1 \downarrow y_2.$$

Graphically:

$$\begin{tikzcd}
x \arrow[r] \arrow[d, "*"'] & y_1 \arrow[d, dashed, "*"] \\
y_2 \arrow[r, dashed, "*"'] & z
\end{tikzcd}$$

Solid arrows represent universal and dashed arrows existential quantification: $\forall x, y_1, y_2.\ y_1 \leftarrow x \xrightarrow{*} y_2 \Rightarrow \exists z.\ y_1 \xrightarrow{*} z \xleftarrow{*} y_2.$

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(1 \Rightarrow 2)$

□

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(1 \Rightarrow 2)$

- Assume → is CR and $y_1 \xleftarrow{*} x \xrightarrow{*} y_2$. Show $y_1 \downarrow y_2$.

$\square$

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(1 \Rightarrow 2)$

- Assume → is CR and $y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2$. Show $y_1 \downarrow y_2$.
- $y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2$ implies $y_1 \overset{*}{\leftrightarrow} y_2$.

$\square$

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(1 \Rightarrow 2)$

- Assume → is CR and $y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2$. Show $y_1 \downarrow y_2$.
- $y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2$ implies $y_1 \overset{*}{\leftrightarrow} y_2$.
- CR implies $y_1 \downarrow y_2$.

□

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(2 \Rightarrow 3)$

□

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. $\rightarrow$ *has the Chursh-Rosser property.*
2. $\rightarrow$ *is confluent.*
3. $\rightarrow$ *is semi-confluent.*

Proof.

$(2 \Rightarrow 3)$

- Semi-confluence is a special case of confluence.

□

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. $\rightarrow$ *has the Chursh-Rosser property.*
2. $\rightarrow$ *is confluent.*
3. $\rightarrow$ *is semi-confluent.*

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$

☐

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$

- Assume → is SC and $x \overset{*}{\leftrightarrow} y$. Show $x \downarrow y$.

☐

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$

- Assume → is SC and $x \overset{*}{\leftrightarrow} y$. Show $x \downarrow y$.
- Induction on the length of the chain $x \overset{*}{\leftrightarrow} y$.

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$

- Assume → is SC and $x \overset{*}{\leftrightarrow} y$. Show $x \downarrow y$.
- Induction on the length of the chain $x \overset{*}{\leftrightarrow} y$.
- Base case: $x = y$. Trivial.

□

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

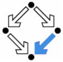
1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$

- Assume → is SC and $x \overset{*}{\leftrightarrow} y$. Show $x \downarrow y$.
- Induction on the length of the chain $x \overset{*}{\leftrightarrow} y$.
- Base case: $x = y$. Trivial.
- Assume $x \overset{*}{\leftrightarrow} y \leftrightarrow y'$. Show $x \downarrow y'$.

□



Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

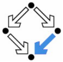
1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$

- Assume → is SC and $x \overset{*}{\leftrightarrow} y$. Show $x \downarrow y$.
- Induction on the length of the chain $x \overset{*}{\leftrightarrow} y$.
- Base case: $x = y$. Trivial.
- Assume $x \overset{*}{\leftrightarrow} y \leftrightarrow y'$. Show $x \downarrow y'$.
- By IH, $x \downarrow y$, i.e. $x \overset{*}{\rightarrow} z \overset{*}{\leftarrow} y$ for some z.

□



Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$ (Cont.)

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. $\rightarrow$ *has the Chursh-Rosser property.*
2. $\rightarrow$ *is confluent.*
3. $\rightarrow$ *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$ (Cont.)

- Show $x \downarrow y'$ by case distinction on $y \leftrightarrow y'$.

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$ (Cont.)

- Show $x \downarrow y'$ by case distinction on $y \leftrightarrow y'$.
- $y \leftarrow y'$: $x \downarrow y'$ follows directly from $x \downarrow y$:

$$\begin{tikzcd} x \arrow[rr, leftrightarrow, "*"] \arrow[rr, phantom, "\text{IH}"', shift right=4] \arrow[dr, dashed, "*"'] & & y \arrow[dl, dashed, "*"] & y' \arrow[l] \\ & {} & & \end{tikzcd}$$

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$ (Cont.)

- Show $x \downarrow y'$ by case distinction on $y \leftrightarrow y'$.

Church-Rosser, Confluence, and Semi-Confluence

Theorem 1.1

The following conditions are equivalent:

1. → *has the Chursh-Rosser property.*
2. → *is confluent.*
3. → *is semi-confluent.*

Proof.

$(3 \Rightarrow 1)$ (Cont.)

- Show $x \downarrow y'$ by case distinction on $y \leftrightarrow y'$.
- $y \to y'$: Semi-confluence implies $z \downarrow y'$ and, hence $x \downarrow y'$:

$$\begin{tikzcd}
x \arrow[rr, leftrightarrow, "*"] \arrow[dr, dashed, "*"'] & {} \arrow[d, phantom, "\text{IH}"] & y \arrow[rr] \arrow[dl, dashed, "*"] & {} \arrow[d, phantom, "\text{SC}"] & y' \arrow[dl, dashed, "*"] \\
& z \arrow[rr, dashed, "*"'] & & {} &
\end{tikzcd}$$

Corollaries

- If → is confluent and $x \overset{*}{\leftrightarrow} y$ then
    1. $x \overset{*}{\rightarrow} y$ if y is in a normal form, and
    2. $x = y$ if both x and y are in a normal form.
- Hence, for confluent relations, convertibility is equivalent to joinability.
- Without termination, joinability can not be decided.

Corollaries

- If $\rightarrow$ is confluent, then every element has at most one normal form ($\mathrm{C} \Longrightarrow \mathrm{UN}$)
- If $\rightarrow$ is normalizing and confluent, then every element has exactly one normal form.

Hence, for confluent and normalizing reductions the notation $x\downarrow$ is well-defined.

Goal-Directed Equivalence Test

Theorem 1.2

If → is confluent and normalizing, then

- *every element* x *has a unique normal form* $x\downarrow$,
- $x \overset{*}{\leftrightarrow} y$ *iff* $x\downarrow = y\downarrow$.

Normalization requires bread-first search for normal forms.

Goal-Directed Equivalence Test

Theorem 1.2

If $\rightarrow$ *is confluent and normalizing, then*

- *every element* x *has a unique normal form* $x\downarrow$*,*
- $x \overset{*}{\leftrightarrow} y$ *iff* $x\downarrow = y\downarrow$*.*

Normalization requires bread-first search for normal forms.

Theorem 1.3

If $\rightarrow$ *is confluent and terminating, then*

- *every element* x *has a unique normal form* $x\downarrow$*,*
- $x \overset{*}{\leftrightarrow} y$ *iff* $x\downarrow = y\downarrow$*.*

Termination permits depth-first search for normal forms.

Confluence and Termination

- How to show confluence and termination of an ARS?

Showing Termination

- Idea: Embedding the reduction into a well-founded order.

Showing Termination

- Idea: Embedding the reduction into a well-founded order.
- Well-founded order $(B, >)$: No infinite descending chain $b_0 > b_1 > b_2 > \cdots$ in B.

Showing Termination

Examples of well-founded orders:

- $(\mathbb{N}, >)$: The set of natural numbers with the standard ordering.
- $(\mathbb{N} \smallsetminus \{0\}, >)$: The set of positive integers where $a > b$ iff $b \mid a$ and $b \neq a$.
- $(\{a, b, c\}^*, >)$: The set of finite words over a fixed alphabet, where $w_1 > w_2$ iff w_2 is a proper substring of w_1.

Showing Termination

Examples of well-founded orders:

- $(\mathbb{N}, >)$: The set of natural numbers with the standard ordering.
- $(\mathbb{N} \smallsetminus \{0\}, >)$: The set of positive integers where $a > b$ iff $b \mid a$ and $b \neq a$.
- $(\{a, b, c\}^*, >)$: The set of finite words over a fixed alphabet, where $w_1 > w_2$ iff w_2 is a proper substring of w_1.

Examples of non-well-founded orders:

- $(\mathbb{Z}, >)$: The set of integers with the standard ordering.
- $(\mathbb{Q}_0^+, >)$: The set of non-negative rationals with the standard ordering.
- $(\{a, b, c\}^*, >)$: The set of finite words over a fixed alphabet, where $>$ is the lexicographic ordering, e.g. $a > ab > abb > \cdots$.

Showing Termination

Theorem 1.4

Let $(A, \to)$ *be an ARS. Then* $\to$ *is terminating iff there exists a well-founded order* $(B, >)$ *and a mapping* $\varphi : A \to B$ *such that*

$$a_1 \to a_2 \text{ implies } \varphi(a_1) > \varphi(a_2).$$

Showing Confluence

Lemma 1.1 (Newman's Lemma)

If → *is terminating and locally confluent, then it is confluent.*

Showing Confluence

Lemma 1.1 (Newman's Lemma)

If → is terminating and locally confluent, then it is confluent.

Proof.

- Use well-founded induction. Let $(A, \to)$ be an ARS. Then WFI is the inference rule:

$$\frac{\forall x \in A.(\forall y \in A.(x \xrightarrow{+} y \Rightarrow P(y)) \Rightarrow P(x))}{\forall x.P(x)} \text{ (WFI)}$$

  where P is some property of elements of A.
- Reads: To prove $P(x)$ for all $x \in A$, try to prove $P(x)$ under the assumption that $P(y)$ holds for all successors y of x.
- Holds when → is terminating.

Showing Confluence

Lemma 1.1 (Newman's Lemma)

If $\rightarrow$ *is terminating and locally confluent, then it is confluent.*

Proof. (Cont.)

Showing Confluence

Lemma 1.1 (Newman's Lemma)

If → is terminating and locally confluent, then it is confluent.

Proof. (Cont.)

- Let P be

$$P(x) = \forall y, z.\ y \xleftarrow{*} x \xrightarrow{*} z \Rightarrow y \downarrow z.$$

Obviously, → is confluent if $P(x)$ holds for all $x \in A$.

Showing Confluence

Lemma 1.1 (Newman's Lemma)

If → is terminating and locally confluent, then it is confluent.

Proof. (Cont.)

- Let P be

  $$P(x) = \forall y, z.\ y \overset{*}{\leftarrow} x \overset{*}{\rightarrow} z \Rightarrow y \downarrow z.$$

  Obviously, → is confluent if $P(x)$ holds for all $x \in A$.
- Show $P(x)$ under the assumption $P(t)$ for all $x \overset{+}{\rightarrow} t$.

Showing Confluence

Lemma 1.1 (Newman's Lemma)

If → is terminating and locally confluent, then it is confluent.

Proof. (Cont.)

- Let P be
$$P(x) = \forall y, z.\ y \overset{*}{\leftarrow} x \overset{*}{\rightarrow} z \Rightarrow y \downarrow z.$$
Obviously, → is confluent if $P(x)$ holds for all $x \in A$.
- Show $P(x)$ under the assumption $P(t)$ for all $x \overset{+}{\rightarrow} t$.
- Fix x, y, z arbitrarily. Assume $y \overset{*}{\leftarrow} x \overset{*}{\rightarrow} z$. Prove $y \downarrow z$.

Showing Confluence

Lemma 1.1 (Newman's Lemma)

If → is terminating and locally confluent, then it is confluent.

Proof. (Cont.)

- Let P be

$$P(x) = \forall y, z.\ y \overset{*}{\leftarrow} x \overset{*}{\rightarrow} z \Rightarrow y \downarrow z.$$

  Obviously, → is confluent if $P(x)$ holds for all $x \in A$.
- Show $P(x)$ under the assumption $P(t)$ for all $x \overset{+}{\rightarrow} t$.
- Fix x, y, z arbitrarily. Assume $y \overset{*}{\leftarrow} x \overset{*}{\rightarrow} z$. Prove $y \downarrow z$.
- Case 1: $x = y$ or $y = x$. Trivial.

Showing Confluence

Lemma 1.1 (Newman's Lemma)

If $\rightarrow$ *is terminating and locally confluent, then it is confluent.*

Proof. (Cont.)

□

Showing Confluence

Lemma 1.1 (Newman's Lemma)

If → *is terminating and locally confluent, then it is confluent.*

Proof. (Cont.)

- Case 2: $x \to y_1 \xrightarrow{*} y$ and $x \to z_1 \xrightarrow{*} z$.

□

Showing Confluence

Lemma 1.1 (Newman's Lemma)

If → is terminating and locally confluent, then it is confluent.

Proof. (Cont.)

- Case 2: $x \to y_1 \overset{*}{\to} y$ and $x \to z_1 \overset{*}{\to} z$.

$$\begin{tikzcd}
x \arrow[r] \arrow[d] \arrow[dr, phantom, "\text{LC}"] & z_1 \arrow[r, "*"] \arrow[d, dashed, "*"'] & z \arrow[dd, dashed, "*"'] \\
y_1 \arrow[r, dashed, "*"] \arrow[d, "*"'] \arrow[dr, phantom, "\text{IH}"] & u \arrow[d, dashed, "*"'] \arrow[r, phantom, "\text{IH}"] & \\
y \arrow[r, dashed, "*"] & v \arrow[r, dashed, "*"] & w
\end{tikzcd}$$