So we need equivalent TRSs with additional properties: termination + confluence.

Termination of a TRS means that \( \rightarrow \) must be well founded.

**Def 3.38 (Well-founded Relation)**

Let \( \rightarrow \) be relation on a set \( M \). The relation \( \rightarrow \) is well-founded iff there is no infinite sequence of elements \( t_0, t_1, \ldots \in M \) with \( t_0 \rightarrow t_1 \rightarrow t_2 \rightarrow \ldots \).

**Ex 3.39**

- \( >_N \) (greater relation on \( N \)) is well founded.
- \( <_N \) is not well founded \((0 < 1 < 5 < 100 < \ldots)\).
- \( > \mathbb{Z} \) \((1 > 0 > -1 > -2 > \ldots)\).
- \( > \mathbb{Q}^+ \) (greater on positive rational numbers) 
  \((1 > \frac{1}{2} > \frac{1}{4} > \frac{1}{8} > \ldots)\).
- \( \triangleright \) (subterm relation) \( f(g(h(x)), y) \triangleright \)
  is well founded.
- \( \triangleright \) is not well founded.
  \( g(h(x)) \triangleright h(x) \triangleright x \)
Our goal is to use TRSs, where every term $s$ can be reduced to a normal form $s_n$.

**Def 3.3.10 (Normal Form)**

Let $\rightarrow$ be a relation on a set $M$. An element $q \in M$ is a normal form iff there is no $q' \in M$ with $q \rightarrow q'$.

An element $q$ is a normal form of $t$ iff $t \rightarrow^{*} q$ and $q$ is a normal form.

If the normal form of $t$ is unique, then $t_N$ denotes the normal form of $t$.

A relation $\rightarrow$ is normalizing iff every element $t$ has (at least) one normal form.

A relation $\rightarrow$ is uniquely normalizing iff every element $t$ has exactly one normal form.

**Lemma 3.3.11. (Well-Foundedness $\implies$ Normalizing)**

Every well-founded relation is normalizing.

**Proof:** If $t \in M$ had no normal form, then $t \rightarrow t_1 \rightarrow t_2 \rightarrow \ldots$ which contradicts the well-foundedness of $\rightarrow$. \hfill $\blacksquare$
**Ex 3.3.13**

(a) \{ b \to a, b \to f(b) \} is normalizing, but not terminating, and not uniquely normalizing. E.g.: \( b \) has the normal forms \( a, f(a), \ldots \)

\[
\begin{align*}
& b \to f(b) \to f(f(b)) \to f^3(b) \to \cdots \\
& a \downarrow \quad f(a) \quad f^2(a) \quad f^3(a)
\end{align*}
\]

(b) \{ b \to a, b \to b \} is uniquely normalizing, but not terminating.

(c) \{ b \to a, b \to c \} is terminating, but not uniquely normalizing.

(d) \{ a \to b, c \to b \} is terminating and uniquely normalizing.

(e) \{ b \to f(b) \} is not normalizing, as \( b \) has no normal form.

**Def 3.3.12. (Termination of TRSs)**

A TRS \( R \) is **terminating** iff \( \rightarrow^R \) is well founded.

A TRS \( R \) is (uniquely) **normalizing** iff \( \rightarrow^R \) is (uniquely) normalizing.

For that reason, we will introduce techniques to prove termination of TRSs in Chapter 4.
Motivation: Word problem, but also many applications in program analysis and verification.

In addition to termination, we want unique normalization. Reason: Otherwise the algorithm for the word problem could return "False" although \( S \equiv t \) holds.

More precisely: We want that \( S \leftrightarrow^* t \) implies that \( S \rightarrow^\ast q \) and \( t \rightarrow^\ast q \) for some term \( q \).

This property was proven for the lambda calculus for the first time by Church and Rosser.

Def 3.3.14. (Church-Rosser Property, Joinability)
Let \( \rightarrow \) be a relation on a set \( M \).

Two elements \( s, t \in M \) are joinable (denoted \( s \downarrow t \)) iff \( s \rightarrow^* q \) \( \in \ast t \) for some \( q \in M \).

The relation \( \rightarrow \) has the Church-Rosser property iff for all \( s, t \in M \): if \( s \leftrightarrow^* t \), then \( s \downarrow t \).

A TRS \( R \) has the Church-Rosser property iff \( \rightarrow^* \) has the Church-Rosser property.

Ex 3.3.15. \( R = \{ b \rightarrow c, b \rightarrow a \} \)
Ex 33.15.  \( R = \{ b \rightarrow c, \ b \rightarrow a \} \)

\( a \leftrightarrow R c \), because \( a \leftrightarrow R b \rightarrow R c \)

But \( a \not\leftrightarrow R c \), since \( a \) and \( c \) are already in normal form.

To check whether a TRS has the CR-property, it is easier to regard a simpler property: confluence. Here, one has to check whether every indeterminism can be resolved again.

\[
\begin{array}{c}
p \\ \downarrow s \\ \downarrow t \\ q \\
\end{array}
\]

if the black arrows hold, do the green arrows hold as well?

Def 33.16. (Confluence)

A relation \( \rightarrow \) on a set \( M \) is confluent iff for all \( p, s, t \in M \):

If \( p \rightarrow s \) and \( p \rightarrow t \),

then there exists a \( q \in M \) such that \( s \rightarrow q \) and \( t \rightarrow q \).

How is confluence related to the CR-property?

Thm 33.17. (CR property \( \implies \) Confluence)

A relation \( \rightarrow \) has the CR property iff \( \rightarrow \) is confluent.
Proof: $\Rightarrow$ \[ \forall P \forall s \forall t \exists s' \in \mathbb{N} \quad s \rightarrow^* t \iff s' \rightarrow^* t \]

\[ \Leftarrow: \text{Let } \rightarrow \text{ be confluent.} \]

Let $s \rightarrow^* t$, i.e., $s \rightarrow^n t$ for some $n \in \mathbb{N}$.

We prove that $s \vdash t$ by induction on $n$.

\textbf{Ind Base: } $n = 0$

$s \rightarrow^0 t \iff s = t \text{ and } s \vdash t$, since $s \rightarrow^0 t$.

\textbf{Ind. Step: } $n > 0$

$s \rightarrow^{n+1} s' \rightarrow t$

by the inductive hypothesis.

\textbf{Case 1: } $t \rightarrow s'$

$s \rightarrow^{n-1} s' \rightarrow t$

Therefore $s \vdash t$.

\textbf{Case 2: } $s' \rightarrow t$

$s \rightarrow^{n-1} s' \rightarrow t$

Therefore $s \vdash t$.

Thus: it suffices to check confluence.
Confluence is needed for programs in order to guarantee that computations have a unique result (i.e., that normal forms are unique).

Lemma 3.3.18. (Confluence means unique normal forms)

(a) If \( \rightarrow \) is confluent, then every object has at most one normal form.

(b) If \( \rightarrow \) is normalizing + confluent, then every object has exactly one normal form (i.e., \( \rightarrow \) is uniquely normalizing).

(c) If \( \rightarrow \) is uniquely normalizing, then \( \rightarrow \) is confluent.

Proof: (a) Assume that \( \overline{t} \) has 2 normal forms \( q_1 \) and \( q_2 \).

\[
\begin{array}{c}
q_1 \xrightarrow{} t \xrightarrow{} q_2 \\
q_1 \xrightarrow{} t \xrightarrow{} q_2 \\
\text{because of} \text{confluence}
\end{array}
\]

Since \( q_1, q_2 \) are normal forms, we have \( q_1 = q = q_2 \).

(b) Follows from (a) by the definition of “normalizing.”

(c) Let \( \rightarrow \) be uniquely normalizing.

Let \( s \vdash P \vdash t \).

\[
\begin{array}{c}
sl \xrightarrow{} \downarrow s \xrightarrow{} \downarrow t \\
sl = tl \quad \{ sl \text{ and } tl \text{ are the unique normal forms of } s \text{ and } t \}
\end{array}
\]

So \( p \) has the normal forms \( sl \) and \( tl \).

By unique normalization, we have \( sl = tl \).

The following theorem states that to check \( s \xrightarrow{}^* t \)

one only has to check whether the normal forms of \( s \) and \( t \).
Theorem 3.3.19 (Checking $\rightarrow^*$ by Normal Forms)

Let $\rightarrow$ be normalizing and confluent. Then: $s \rightarrow^* t \iff s \downarrow = t \downarrow$.

Proof:

\[\begin{align*}
\Rightarrow: & \quad s \rightarrow^* t \Rightarrow s \downarrow t \Rightarrow s \downarrow = t \downarrow \\
\Rightarrow: & \quad s \downarrow = t \downarrow \Rightarrow s \downarrow t \Rightarrow s \rightarrow^* t
\end{align*}\]

Since $\rightarrow$ is confluent, which is equivalent to the CR-property (Theorem 3.3.17).

If $\rightarrow$ is only normalizing (but not well founded), then it could be difficult to find the normal form of a term $s$.

$s \rightarrow^* s \downarrow$, but $s$ might also start infinite rewrite sequences.

Therefore, we require that $\rightarrow$ should be well-founded and confluent.

Definition 3.3.20 (Convergence of TRSs)

A TRS is \textit{convergent} iff it is terminating and confluent.

Example 3.3.21

Convergent TRSs correspond to an interpreter that evaluates expressions to a result.

The plus-TRS is convergent. It can be used to evaluate expressions with "plus".
\[ \text{plus}(3, 1) = \text{plus}(s(s(0)), s(0)) \]

Evaluation will terminate with the unique result
\[ 3 \overset{\eta}{=} s(s(s(0))) \]

Convergent TRSs can not only be used to compute results, but also to prove equations:

**Algorithm WORD-PROBLEM (S, S, t)**

**Thm 3.3.22.** (Correctness of Alg. WORD-PROBLEM)
(a) The alg. WORD-PROBLEM terminates.
(b) If \( S \) is equivalent to \( E \), then the alg. WP is correct.
(c) If a set of equations \( E \) has an equivalent convergent TRS, then the word problem for \( E \) is decidable.

**Proof:**
(a) Termination follows from termination of \( S \).
(b) \( S \models t \) iff \( S \overset{\eta}{=} s(t) \) iff \( S \overset{\eta}{=} s(t) \) iff \( s(t) \models s(t) \).

**Thm 3.3.19**

**Proof: Birkhoff**

**Thm 3.3.19**

E and S are equivalent

(c) follows from the fact that WP is a decision procedure.
(This alg. can be executed automatically. An algorithm for matching will be presented in Section 5.)

**Ex. 33.23.** Group Example
\[ E = \{ t(x, t(y, z)) \overset{\eta}{=} t(t(x, y), z), t(x, e) = x, t(x, i(x)) = e \} \]

Orienting these equations yields a TRS with 3 rules. Using algorithm WORD-PROBLEM to check
i(i(n)) \equiv n \quad \text{yields "False"}

Problem: The 3-rule TRS is not confluent:

\[
f(x, f(y, i(y))) \
\rightarrow \leftarrow \quad f(f(x, i(y)), i(y)) \rightarrow f(x, e)
\]

2 different normal forms

Solution: Extend the TRS by additional rules in order to make it confluent: "Completion of TRSs".

General idea: Whenever there is an indeterminism that can't be joined, add this indeterminism as an additional rule.

Here: add the rule \( f(f(x, i(y)), i(y)) \rightarrow x \)

We will later introduce techniques to complete TRSs automatically (i.e., the 10-rule TRS is generated automatically from the 3 group axioms).

The 10-rule TRS is convergent \( \Rightarrow \) it is a decision procedure for groups.

E.g.: check whether \( i(f(i(n), f(v, u))) \equiv \epsilon f(i(n), f(i(v), u)) \)

So we now have a decision procedure for equations about groups.

So our goal is to have decision procedures for statements about algorithms or data structures.
Moreover, we want to synthesize these decision procedures automatically: see flowchart.

Tasks

- Termination of TRSs (Sect. 4)
- Confluence (Sect. 5)
- Completion (Sect. 6)