4.1.11 Theorem (Birkhoff's Theorem)

Let $\mathcal{X}$ be a countably infinite set of variables, let E be a set of (universally quantified) equations. Then the following properties are equivalent for all $s, t \in T_S(\Sigma, \mathcal{X})$:

1. $s \leftrightarrow_E^* t$.
2. $E \Rightarrow_E^* s \approx t$ is derivable.
3. $s \approx_E t$, i.e., $E \models \forall \vec{x}(s \approx t)$.
4. $\mathcal{I}_E \models \forall \vec{x}(s \approx t)$.

By Theorem 4.1.11 the semantics of E and $\leftrightarrow_E^*$ conincide. In order to decide $\leftrightarrow_E^*$ we need to turn $\rightarrow_E^*$ in a confluent and terminating relation.

If $\leftrightarrow_E^*$ is terminating then confluence is equivalent to local confluence, see Newman's Lemma, Lemma 1.6.6. Local confluence is the following problem for TRS: if $t_1 \; {}_E\!\!\leftarrow t_0 \rightarrow_E t_2$, does there exist a term s so that $t_1 \rightarrow_E^* s \; {}_E^*\!\!\leftarrow t_2$?

If the two rewrite steps happen in different subtrees (disjoint redexes) then a repitition of the respective other step yields the common term s.

If the two rewrite steps happen below each other (overlap at or below a variable position) again a repetition of the respective other step yields the common term s.

If the left-hand sides of the two rules overlap at a non-variable position there is no ovious way to generate s.

More technically two rewrite rules $l_1 \to r_1$ and $l_2 \to r_2$ overlap if there exist some non-variable subterm $l_1|_p$ such that l_2 and $l_1|_p$ have a common instance $(l_1|_p)\sigma_1 = l_2\sigma_2$. If the two rewrite rules do not have common variables, then only a single substitution is necessary, the mgu σ of $(l_1|_p)$ and l_2.

4.2.1 Definition (Critical Pair)

Let $l_i \to r_i$ ($i = 1, 2$) be two rewrite rules in a TRS R whithout common variables, i.e., $\mathrm{vars}(l_1) \cap \mathrm{vars}(l_2) = \emptyset$. Let $p \in \mathrm{pos}(l_1)$ be a position so that $l_1|_p$ is not a variable and σ is an mgu of $l_1|_p$ and l_2. Then $r_1\sigma \leftarrow l_1\sigma \to (l_1\sigma)[r_2\sigma]_p$.

$\langle r_1\sigma, (l_1\sigma)[r_2\sigma]_p\rangle$ is called a *critical pair* of R.

The critical pair is *joinable* (or: converges), if $r_1\sigma \downarrow_R (l_1\sigma)[r_2\sigma]_p$.

4.2.2 Theorem ("Critical Pair Theorem")

A TRS R is locally confluent iff all its critical pairs are joinable.

Knuth-Bendix Completion (KBC)

Given a set E of equations, the goal of Knuth-Bendix completion is to transform E into an equivalent convergent set R of rewrite rules. If R is finite this yields a decision procedure for E.

For ensuring termination the calculus fixes a reduction ordering $\succ$ and constructs R in such a way that $\rightarrow_R \subseteq \succ$, i.e., $l \succ r$ for every $l \rightarrow r \in R$.

For ensuring confluence the calculus checks whether all critical pairs are joinable.

The completion procedure itself is presented as a set of abstract rewrite rules working on a pair of equations E and rules R:
$(E_0;R_0) \Rightarrow_{\text{KBC}} (E_1;R_1) \Rightarrow_{\text{KBC}} (E_1;R_2) \Rightarrow_{\text{KBC}} \ldots$.

The initial state is $(E_0, \emptyset)$ where $E = E_0$ contains the input equations.

If $\Rightarrow_{\text{KBC}}$ successfully terminates then E is empty and R is the convergent rewrite system for E_0.

For each step $(E; R) \Rightarrow_{\text{KBC}} (E'; R')$ the equational theories of $E \cup R$ and $E' \cup R'$ agree: $\approx_{E\cup R} = \approx_{E'\cup R'}$. By $\text{cp}(R)$ I denote the set of critical pairs between rules in R.

Orient $(E \uplus \{s \dot{\approx} t\}; R) \Rightarrow_{\text{KBC}} (E; R \cup \{s \rightarrow t\})$
if $s \succ t$

Delete $(E \uplus \{s \approx s\}; R) \Rightarrow_{\text{KBC}} (E; R)$

Deduce $(E; R) \Rightarrow_{\text{KBC}} (E \cup \{s \approx t\}; R)$
if $\langle s, t \rangle \in \text{cp}(R)$

Simplify-Eq $(E \uplus \{s \dot{\approx} t\}; R) \Rightarrow_{\mathrm{KBC}} (E \cup \{u \approx t\}; R)$

if $s \rightarrow_R u$

R-Simplify-Rule $(E; R \uplus \{s \rightarrow t\}) \Rightarrow_{\mathrm{KBC}} (E; R \cup \{s \rightarrow u\})$

if $t \rightarrow_R u$

L-Simplify-Rule $(E; R \uplus \{s \rightarrow t\}) \Rightarrow_{\mathrm{KBC}} (E \cup \{u \approx t\}; R)$

if $s \rightarrow_R u$ using a rule $l \rightarrow r \in R$ so that $s \sqsupset l$, see below.

Trivial equations cannot be oriented and since they are not needed they can be deleted by the Delete rule.

The rule Deduce turns critical pairs between rules in R into additional equations. Note that if $\langle s, t\rangle \in \mathrm{cp}(R)$ then $s_R \leftarrow u \rightarrow_R t$ and hence $R \models s \approx t$.

The simplification rules are not needed but serve as reduction rules, removing redundancy from the state. Simplification of the left-hand side may influence orientability and orientation of the result. Therefore, it yields an equation. For technical reasons, the left-hand side of $s \rightarrow t$ may only be simplified using a rule $l \rightarrow r$, if $l \rightarrow r$ cannot be simplified using $s \rightarrow t$, that is, if $s \sqsupset l$, where the *encompassment quasi-ordering* $\gtrsim$ is defined by $s \gtrsim l$ if $s|_p = l\sigma$ for some p and σ and $\sqsupset = \gtrsim \setminus \lesssim$ is the strict part of $\gtrsim$.

4.4.4 Proposition (Knuth-Bendix Completion Correctness)

If the completion procedure on a set of equations E is run, different things can happen:

1. A state where no more inference rules are applicable is reached and E is not empty. $\Rightarrow$ Failure (try again with another ordering?)
2. A state where E is empty is reached and all critical pairs between the rules in the current R have been checked.
3. The procedure runs forever.

4.4.5 Definition (Run)

A (finite or infinite) sequence $(E_0; R_0) \Rightarrow_{KBC} (E_1; R_1) \Rightarrow_{KBC} (E_2; R_2) \Rightarrow_{KBC} \ldots$ with $R_0 = \emptyset$ is called a *run* of the completion procedure with input E_0 and $\succ$. For a run, $E_\infty = \bigcup_{i\geq 0} E_i$ and $R_\infty = \bigcup_{i\geq 0} R_i$.

4.4.6 Definition (Persistent Equations)

The sets of *persistent equations of rules* of the run are $E_* = \bigcup_{i\geq 0} \bigcap_{j\geq i} E_j$ and $R_* = \bigcup_{i\geq 0} \bigcap_{j\geq i} R_j$.

Note: If the run is finite and ends with E_n, R_n then $E_* = E_n$ and $R_* = R_n$.

4.4.7 Definition (Fair Run)

A run is called *fair* if $\mathrm{CP}(R_*) \subseteq E_\infty$ (i.e., if every critical pair between persisting rules is computed at some step of the derivation).

4.4.10 Theorem (KBC Soundness)

Let $(E_0; R_0) \Rightarrow_{KBC} (E_1; R_1) \Rightarrow_{KBC} (E_2; R_2) \Rightarrow_{KBC} \ldots$ be a fair run and let R_0 and E_* be empty. Then

1. every proof in $E_\infty \cup R_\infty$ is equivalent to a rewrite proof in R_*,
2. R_* is equivalent to E_0 and
3. R_* is convergent.

Complexity

3.15.2 Theorem (Equational Logic Validity is Undecidable)

Validity of an equation modulo a set of equations is undecidable.

(Proof Scetch) Given a PCP with word lists $(u_1, \ldots, u_n)$ and $(v_1, \ldots, v_n)$ over alphabet $\{a, b\}$, it is represented by two unary functions g_a and g_b, constants ϵ, c, d, and a binary function f_R , all over some sort S. Then a word pair u_i, v_i is encoded by the equation $f_R(u_i(x), v_i(y)) \approx f_R(x, y)$ the start state with the empty word by equation $f_R(\epsilon, \epsilon) \approx d$ and the final state identifying two equal words different from ϵ by the equations $f_R(g_a(x), g_a(x)) \approx c, \quad f_R(g_b(x), g_b(x)) \approx c$. I call the set of these equations E. Now the PCP over the two word lists has a solution iff $E \models c \approx d$.

4.4.11 Corollary (KBC Termination)

Termination of $\Rightarrow_{KBC}$ is undecidable for some given finite set of equations E.

(Proof Scetch) Using exactly the construction of Theorem 3.15.2 it remains to be shown that all computed critical pairs can be oriented. Critical pairs corresponding to the search for a PCP solution result in equations $f_R(u(x), v(y)) \approx f_R(u'(x), v'(y))$ or $f_R(u'(x), v'(x)) \approx c$. By chosing an appropriate ordering, all these equations can be oriented. Thus $\Rightarrow_{KBC}$ does not produce any unorientable equations. The rest follows from Theorem 3.15.2.