3.11 Orderings

Propositional superposition is based on an ordering on the propositional variables, Section 2.7. The ordering is total and well-founded. Basically, propositional variables correspond to ground atoms in first-order logic. This section generalizes the ideas of the propositional superposition ordering to first-order logic. In first-order logic the ordering has to also consider terms and variables and operations on terms like the application of a substitution.

Definition 3.11.1 (Σ-Operation Compatible Relation)**.** A binary relation $\sqsupset$ over $T(\Sigma, \mathcal{X})$ is called *compatible with* Σ*-operations,* if $s \sqsupset s'$ implies $f(t_1, \ldots, s, \ldots, t_n) \sqsupset f(t_1, \ldots, s', \ldots, t_n)$ for all $f \in \Omega$ and $s, s', t_i \in T(\Sigma, \mathcal{X})$.

Lemma 3.11.2 (Σ-Operation Compatible Relation)**.** A relation $\sqsupset$ is compatible with Σ-operations iff $s \sqsupset s'$ implies $t[s]_p \sqsupset t[s']_p$ for all $s, s', t \in T(\Sigma, \mathcal{X})$ and $p \in pos(t)$.

In the literature *compatible with* Σ*-operations* is sometimes also called *compatible with contexts.*

Definition 3.11.3 (Substitution Stable Relation, Rewrite Relation)**.** A binary relation $\sqsupset$ over $T(\Sigma, \mathcal{X})$ is called *stable under substitutions*, if $s \sqsupset s'$ implies $s\sigma \sqsupset s'\sigma$ for all $s, s' \in T(\Sigma, \mathcal{X})$ and substitutions σ. A binary relation $\sqsupset$ is called a *rewrite relation*, if it is compatible with Σ-operations and stable under substitutions.

A *rewrite ordering* is then an ordering that is a rewrite relation.

Definition 3.11.4 (Subterm Ordering)**.** The *proper subterm ordering* $s > t$ is defined by $s > t$ iff $s|_p = t$ for some position $p \neq \epsilon$ of s.

Definition 3.11.5 (Simplification Ordering)**.** A rewrite ordering $\succ$ over $T(\Sigma, \mathcal{X})$ is called *simplification ordering*, if it enjoys the *subterm property* $s \succ t$ implies $s > t$ for all $s, t \in T(\Sigma, \mathcal{X})$ of the same sort.

Definition 3.11.6 (Lexicographical Path Ordering (LPO))**.** Let $\Sigma = (\mathcal{S}, \Omega, \Pi)$ be a signature and let $\succ$ be a strict partial ordering on operator symbols in Ω, called *precedence*. The *lexicographical path ordering* $\succ_{lpo}$ on $T(\Sigma, \mathcal{X})$ is defined as follows: if s, t are terms in $T_S(\Sigma, \mathcal{X})$ then $s \succ_{lpo} t$ iff

1. $t = x \in \mathcal{X}$, $x \in vars(s)$ and $s \neq t$ or
2. $s = f(s_1, \ldots, s_n)$, $t = g(t_1, \ldots, t_m)$ and
   (a) $s_i \succeq_{lpo} t$ for some $i \in \{1, \ldots, n\}$ or
   (b) $f \succ g$ and $s \succ_{lpo} t_j$ for every $j \in \{1, \ldots, m\}$ or
   (c) $f = g$, $s \succ_{lpo} t_j$ for every $j \in \{1, \ldots, m\}$ and $(s_1, \ldots, s_n)(\succ_{lpo})_{lex}(t_1, \ldots, t_m)$.

Theorem 3.11.7 (LPO Properties)**.** 1. The LPO is a rewrite ordering.

2. LPO enjoys the subterm property, hence is a simplification ordering.
3. If the precedence $\succ$ is total on Ω then $\succ_{lpo}$ is total on the set of ground terms $T(\Sigma)$.
4. If Ω is finite then $\succ_{lpo}$ is well-founded.

Example 3.11.8. Consider the terms $g(x)$, $g(y)$, $g(g(a))$, $g(b)$, $g(a)$, b, a. With respect to the precedence $g \succ b \succ a$ the ordering on the ground terms is $g(g(a)) \succ_{lpo} g(b) \succ_{lpo} g(a) \succ_{lpo} b \succ_{lpo} a$. The terms $g(x)$ and $g(y)$ are not comparable. Note that the terms $g(g(a))$, $g(b)$, $g(a)$ are all instances of both $g(x)$ and $g(y)$.

With respect to the precedence $b \succ a \succ g$ the ordering on the ground terms is $g(b) \succ_{lpo} b \succ_{lpo} g(g(a)) \succ_{lpo} g(a) \succ_{lpo} a$.

Definition 3.11.9 (The Knuth-Bendix Ordering)**.** Let $\Sigma = (\mathcal{S}, \Omega, \Pi)$ be a finite signature, let $\succ$ be a strict partial ordering *("precedence")* on Ω, let $w : \Omega \cup \mathcal{X} \to \mathbb{R}^+$ be a *weight function*, so that the following admissibility condition is satisfied: $w(x) = w_0 \in \mathbb{R}^+$ for all variables $x \in \mathcal{X}$; $w(c) \geq w_0$ for all constants $c \in \Omega$.

Then, the weight function w can be extended to terms recursively:

$$w(f(t_1, \ldots, t_n)) = w(f) + \sum_{1 \leq i \leq n} w(t_i)$$

or alternatively

$$\sum w(t) = \sum_{x \in vars(t)} w(x) \cdot \#(x,t) + \sum_{f \in \Omega} w(f) \cdot \#(f,t)$$

where $\#(a,t)$ is the number of occurrences of a in t.

The *Knuth-Bendix ordering* $\succ_{kbo}$ on $T(\Sigma, \mathcal{X})$ induced by $\succ$ and admissible w is defined by: $s \succ_{kbo} t$ iff

1. $\#(x,s) \geq \#(x,t)$ for all variables x and $w(s) > w(t)$, or
2. $\#(x,s) \geq \#(x,t)$ for all variables x, $w(s) = w(t)$, and
   (a) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and $f \succ g$, or
   (b) $s = f(s_1, \ldots, s_m)$, $t = f(t_1, \ldots, t_m)$, and $(s_1, \ldots, s_m)(\succ_{kbo})_{lex}(t_1, \ldots, t_m)$.

Theorem 3.11.10 (KBO Properties)**.** 1. The KBO is a rewrite ordering.

2. KBO enjoys the subterm property, hence is a simplification ordering.
3. If the precedence $\succ$ is total on Ω then $\succ_{kbo}$ is total on the set of ground terms $T(\Sigma)$.
4. If Ω is finite then $\succ_{kbo}$ is well-founded.

The KBO ordering can be extended to contain unary function symbols with weight zero. This was motivated by completion of the group axioms, see Chapter 4.

Definition 3.11.11 (The Knuth-Bendix Ordering Extended)**.** The additional requirements added to Definition 3.11.9 are

1. Extend w to $w : \Omega \cup \mathcal{X} \to \mathbb{R}_0^+$
2. If $w(f) = 0$ for some $f \in \Omega$ with arity$(f) = 1$, then $f \succeq g$ for all $g \in \Omega$.
3. As a first case to the disjunction of 3.11.9-2.
   (a') $t = x$, $s = f^n(x)$ for some $n \geq 1$

The LPO ordering as well as the KBO ordering can be extended to atoms in a straightforward way. The precedence $\succ$ is extended to Π. For LPO atoms are then compared according to Definition 3.11.6-2. For KBO the weight function w is also extended to atoms by giving predicates a non-zero positive weight and then atoms are compared according to terms.

Actually, since atoms are never substituted for variables in first-order logic, an alternative to the above would be to first compare the predicate symbols and let $\succ$ decide the ordering. Only if the atoms share the same predicate symbol, the argument terms are considered, e.g., in a lexicographic way and are then compared with respect to KBO or LPO, respectively.

3.12 First-Order Ground Superposition

Propositional clauses and ground clauses are essentially the same, as long as equational atoms are not considered. This section deals only with ground clauses and recalls mostly the material from Section 2.7 for first-order ground clauses. The main difference is that the atom ordering is more complicated, see Section 3.11. Let N be a possibly infinite set of ground clauses.

Definition 3.12.1 (Ground Clause Ordering)**.** Let $\prec$ be a strict rewrite ordering total on ground terms and ground atoms. Then $\prec$ can be lifted to a total ordering $\prec_L$ on literals by its multiset extension $\prec_{\text{mul}}$ where a positive literal $P(t_1,\ldots,t_n)$ is mapped to the multiset $\{P(t_1,\ldots,t_n)\}$ and a negative literal $\neg P(t_1,\ldots,t_n)$ to the multiset $\{P(t_1,\ldots,t_n), P(t_1,\ldots,t_n)\}$. The ordering $\prec_L$ is further lifted to a total ordering on clauses $\prec_C$ by considering the multiset extension of $\prec_L$ for clauses.

Proposition 3.12.2 (Properties of the Ground Clause Ordering)**.**

1. The orderings on literals and clauses are total and well-founded.
2. Let C and D be clauses with $P(t_1,\ldots,t_n) = \text{atom}(\max(C))$, $Q(s_1,\ldots,s_m) = \text{atom}(\max(D))$, where $\max(C)$ denotes the maximal literal in C.

(a) If $Q(s_1,\dots,s_m) \prec_L P(t_1,\dots,t_n)$ then $D \prec_C C$.

(b) If $P(t_1,\dots,t_n) = Q(s_1,\dots,s_m)$, $P(t_1,\dots,t_n)$ occurs negatively in C but only positively in D, then $D \prec_C C$.

Eventually, as I did for propositional logic, I overload $\prec$ with $\prec_L$ and $\prec_C$. So if $\prec$ is applied to literals it denotes $\prec_L$, if it is applied to clauses, it denotes $\prec_C$. Note that $\prec$ is a total ordering on literals and clauses as well. For superposition, inferences are restricted to maximal literals with respect to $\prec$. For a clause set N, I define $N^{\prec C} = \{D \in N \mid D \prec C\}$.

Definition 3.12.3 (Abstract Redundancy)**.** A ground clause C is *redundant* with respect to a set of ground clauses N if $N^{\prec C} \models C$.

Tautologies are redundant. Subsumed clauses are redundant if $\subseteq$ is strict. Duplicate clauses are anyway eliminated quietly because the calculus operates on sets of clauses.

Note that for finite N, and any $C \in N$ redundancy $N^{\prec C} \models C$ can be decided but is as hard as testing unsatisfiability for a clause set N. So the goal is to invent redundancy notions that can be efficiently decided and that are useful.

Definition 3.12.4 (Selection Function)**.** The selection function sel maps clauses to one of its negative literals or $\bot$. If $\text{sel}(C) = \neg P(t_1,\dots,t_n)$ then $\neg P(t_1,\dots,t_n)$ is called *selected* in C. If $\text{sel}(C) = \bot$ then no literal in C is *selected*.

The selection function is, in addition to the ordering, a further means to restrict superposition inferences. If a negative literal is selected in a clause, any superposition inference must be on the selected literal.

Definition 3.12.5 (Partial Model Construction)**.** Given a clause set N, an ordering $\prec$, and a selection function sel the (partial) model $N_\mathcal{I}$ for N is inductively constructed as follows:

$$\begin{array}{lcl}
N_C & := & \bigcup_{D \prec C} \delta_D \\[1ex]
\delta_D & := & \left\{\begin{array}{ll} \{P(t_1,\dots,t_n)\} & \text{if } D = D' \vee P(t_1,\dots,t_n), P(t_1,\dots,t_n) \text{ strictly} \\ & \text{maximal, } \text{sel}(D) = \bot \text{ and } N_D \not\models D \\ \emptyset & \text{otherwise} \end{array}\right. \\
N_\mathcal{I} & := & \bigcup_{C \in N} \delta_C
\end{array}$$

Clauses C with $\delta_C \neq \emptyset$ are called *productive.*

Proposition 3.12.6 (Properties of the Model Operator)**.** Some properties of the partial model construction.

1. For every D with $(C \vee \neg P(t_1,\dots,t_n)) \prec D$ we have $\delta_D \neq \{P(t_1,\dots,t_n)\}$.
2. If $\delta_C = \{P(t_1,\dots,t_n)\}$ then $N_C \cup \delta_C \models C$.