Chapter 2

Mathematical Preliminary

As the foundations of hierarchically sorted inductive logic programming, this chapter provides a concise summary of logic programs, generalization of logic programs, and ψ-terms.

2.1 Logic Programs

2.1.1 Syntax

Definition 1 (Signature) *A signature $\Sigma = (\mathcal{F}, \mathcal{P}, \mathcal{C})$ consists of:*

- *A set of function symbols $\mathcal{F}$.*
- *A set of predicate symbols $\mathcal{P}$.*
- *A set of constant symbols $\mathcal{C}$.*

Definition 2 (Terms)

- *A constant symbol $c \in \mathcal{C}$ is a term.*
- *A variable symbol $X \in \mathcal{V}$ is a term.*
- *If $f^{(n)}$ $(1 \leq n)$ is a function symbol of arity n and $t_1, ..., t_n$ are terms, $f^{(n)}(t_1, ..., t_n)$ is a term.*

Definition 3 (Atomic Formulae) *If $p^{(n)}$ is a predicate symbol of arity n and $t_1, ..., t_n$ are terms, $p^{(n)}(t_1, ..., t_n)$ is an atomic formula.*

An atomic formula is also called an *atom.*

Definition 4 (Literals) *Literals consist of positive literals and negative literals.*

- *An atomic formula is a positive literal.*
- *If P is an atomic formula, $\neg P$ is a negative literal.*

Definition 5 (Ground Terms and Ground Literals) *A ground term is a term that contains no variables. A ground literal is a literal that contains no variables.*

Definition 6 (Clauses) *If $L_1, ..., L_n$ are literals, a set of literals $\{L_1, ..., L_n\}$ is a clause.*

Given positive literals $H_1, ..., H_m$ $(m \geq 0)$ and negative literals $\neg B_1, ..., \neg B_n$ $(n \geq 0)$, another notation of the clause $\{H_1, ..., H_m, \neg B_1, ..., \neg B_n\}$ is

$$H_1, ..., H_m :- B_1, ..., B_n.$$

The left hand side of :– is called the *head* and the right hand side of :– is called the *body* of a clause.

Definition 7 (Horn Clauses) *If $H_1, ..., H_m$ are positive literals with $0 \leq m \leq 1$ and $\neg B_1, ..., \neg B_n$ are negative literals with $n \geq 0$, the set of literals $\{H_1, ..., H_m, \neg B_1, ..., \neg B_n\}$ is a Horn clause.*

Definition 8 (Definite Clauses) *If H is a positive literal and $\neg B_1, ..., \neg B_n$ $(n \geq 0)$ are negative literals, the clause $\{H, \neg B_1, ..., \neg B_n\}$ is a definite clause.*

A definite clause is also called a *program clause.*

Definition 9 (Unit Clauses) *If H is a positive literal, the clause $\{H\}$ is a unit clause.*

Definition 10 (Logic Programs) *A logic program is a finite set of definite clauses.*

Definition 11 (Goal Clauses) *If $\neg B_1, ..., \neg B_n$ $(n \geq 1)$ are negative literals, the clause $\{\neg B_1, ..., \neg B_n\}$ is a goal clause.*

2.1.2 Semantics

Definition 12 (Structure) *Given a signature $\Sigma = \langle \mathcal{F}, \mathcal{P}, \mathcal{C} \rangle$, a structure $M = \langle D, I \rangle$ satisfies the following conditions:*

- *A non-empty set D, called a domain.*
- *A function I s.t.*
  - *If $c \in \mathcal{C}$, $I(c) \in D$.*
  - *If $f^{\langle n \rangle} \in \mathcal{F}$, $I(f^{\langle n \rangle}) : D^n \rightarrow D$.*
  - *If $p^{\langle n \rangle} \in \mathcal{P}$, $I(p^{\langle n \rangle}) \subseteq D^n$.*

Definition 13 (Variable Assignment) *A variable assignment is a function $\sigma : \mathcal{V} \rightarrow \mathcal{D}$.*

Definition 14 (Interpretation) *An interpretation $\mathcal{I}$ is defined as $\mathcal{I} = \langle M, \sigma \rangle$, where M is a structure and σ is a variable assignment.*

Definition 15 (Term Assignment) *Given an interpretation $\mathcal{I} = \langle M, \sigma \rangle$ and a term t, the term assignment $[\![]\!]_\sigma$ of t is given by:*

- *If $t \in \mathcal{C}$, $[\![t]\!]_\sigma = I(t)$.*
- *If $t \in \mathcal{V}$, $[\![t]\!]_\sigma = \sigma(t)$.*
- *If t has the form $f^{\langle n \rangle}(t_1, ..., t_n)$, $[\![t]\!]_\sigma = I(f^{\langle n \rangle})([\![t_1]\!]_\sigma, ..., [\![t_n]\!]_\sigma)$.*

Definition 16 (Satisfaction Relation) *Given a signature $\Sigma = \langle \mathcal{F}, \mathcal{P}, \mathcal{C} \rangle$ and an interpretation $\mathcal{I}$, the satisfaction relation $\models_\mathcal{I}$ is defined as follows.*

- $\models_\mathcal{I} p^{\langle n \rangle}(t_1, ..., t_n)$ *iff* $([\![t_1]\!]_\sigma, ..., [\![t_n]\!]_\sigma) \in I(p^{\langle n \rangle})$.
- $\models_\mathcal{I} \neg F$ *iff* $\not\models_\mathcal{I} F$ *does not hold.*
- $\models_\mathcal{I} \{L_1, ..., L_n\}$ *iff $L_1, ..., L_n$ are literals and for some i,* $\models_\mathcal{I} L_i$.
- $\models_\mathcal{I} \{C_1, ..., C_n\}$ *iff $C_1, ..., C_n$ are clauses and for all i,* $\models_\mathcal{I} C_i$.

Definition 17 (Model) *An interpretation $\mathcal{I}$ is a model of a literal, a clause, or a set of clauses E iff $\models_{\mathcal{I}} E$.*

Definition 18 (Logical Entailment) *Given T is a set of clauses and E is a literal, a clause, or a set of clauses, $T \models E$ iff for every interpretation $\mathcal{I}$, if $\models_{\mathcal{I}} T$ then $\models_{\mathcal{I}} E$.*

We read $T \models E$ as E is a logical entailment of T.

Definition 19 (Herbrand Universe) *Given a signature $\Sigma = \langle \mathcal{F}, \mathcal{P}, \mathcal{C} \rangle$, The Herbrand universe HU for Σ is the set of all ground terms that can be formed using the constants in $\mathcal{C}$ and the function symbols in $\mathcal{F}$. If $\mathcal{C}$ is empty, we add c to $\mathcal{C}$, where c is a new constant, and form the Herbrand universe from $\mathcal{C}$.*

Definition 20 (Herbrand Base) *A Herbrand base HB is*

$$HB = \{p^{(n)}(t_1, ..., t_n) \mid p^{(n)} \in \mathcal{P} \text{ and } t_1, ..., t_n \in HU\}.$$

Definition 21 (Herbrand Interpretation) *Given $M = \langle D, I \rangle$ and an interpretation $\mathcal{I} = \langle M, \sigma \rangle$, an interpretation $\mathcal{I}$ is a Herbrand interpretation, iff the following holds:*

- *D is a Herbrand universe HU.*
- *For all constant $c \in \mathcal{C}$, $I(c) = c$.*
- *For all function $f^{(n)} \in \mathcal{F}$, $I(f^{(n)}) : HU^n \rightarrow HU$ s.t. $I(f^{(n)})(t_1, ..., t_n) = f(t_1, ..., t_n)$.*

Definition 22 (Herbrand Model) *Let T be a set of clauses. A Herbrand interpretation $\mathcal{I}$ is a Herbrand model iff $\mathcal{I}$ is a mode of T.*

Given a Herbrand model $\mathcal{I}$, let $\mathcal{I}' = \{P \in HB \mid \models_{\mathcal{I}} P\}$. Since any Herbrand model $\mathcal{I}$ can be regarded as the subset of the Herbrand base $\mathcal{I}'$, hereafter we treat Herbrand models $\mathcal{I}$ and $\mathcal{I}'$ identically.

Definition 23 (Least Herbrand Model) *Let $\mathcal{I}_1$,...,$\mathcal{I}_n$ be all of the Herbrand models of a set of clauses T. The least Herbrand model $M(T) = \cap_i \mathcal{I}_i$.*

Definition 24 (Empirical Content) *Given a set of clauses T and a predicate name p, an empirical content $Q(T)$ is a set of atoms with the predicate name of p in the least Herbrand model $M(T)$.*

2.1.3 Deduction

Definition 25 (Substitution) *A substitution is the form* $\{V_1/T_1, ..., V_n/T_n\}$, *where* $V_1, ..., V_n$ *are pairwise distinct variables,* $T_1, ..., T_n$ *are terms, and* V_i *and* T_i *are distinct.*

Definition 26 (Instantiation) *Let a substitution* $\theta = \{V_1/T_1, ..., V_n/T_n\}$. *Let* E *be a term, a literal, or a clause.* $E\theta$ *is the result of replacing all occurrences* V_i *with the term* T_i *for all* i *simultaneously.*

Definition 27 (Deduction Relation) *Given sets of definite clauses* T *and* D, *the deduction relation* $T \vdash D$ *holds iff* D *is proven from* T *by a resolution procedure based on Robinson's resolution principle [63].*

The details of resolution procedures can be found in [32].

Given a set of clauses T and a set of unit clauses $\{\{G_1\}, ..., \{G_n\}\}$, we write $T \vdash \{\{G_1\}, ..., \{G_n\}\}$ as $T \vdash G_1, ..., G_n$ for a lighter notation.

Definition 28 (Soundness) *Let* T *and* D *be sets of clauses. A resolution procedure for the deduction relation* $\vdash$ *is sound iff this condition hosts: if* $T \vdash D$ *then* $T \models D$.

Definition 29 (Completeness) *Let* T *and* D *be set of clauses. A resolution procedure for the deduction relation* $\vdash$ *is complete iff this condition hosts: if* $T \models D$ *then* $T \vdash D$.

The SLD resolution [32], one of the resolution procedures for Horn clauses based on Robinson's resolution principle, has the following property.

Theorem 1 (Correctness of SLD Resolution) *[32] The SLD resolution for Horn clauses is complete and sound.*

2.2 Generalization of Logic Programs

2.2.1 Least General Generalization

This section introduces Plotkin's *least general generalization (lgg)* according to [43].

Definition 30 (Ordering) *Let W_1 and W_2 be two terms or two atoms. $W_1 \leq W_2$ iff $W_1\theta = W_2$ for some substitution θ. Let C_1 and C_2 be clauses. $C_1 \leq C_2$ iff $C_1\theta \subseteq C_2$ for some substitution θ.*

We read $L_1 \leq L_2$ as meaning L_1 is more general than L_2 or L_2 is more specific than L_1. If we are ordering only Horn clauses, we introduce $\perp$ as the most specific clause with respect to $\leq$.

Example 2 (Ordering of Clauses) *Suppose that we have the following two clauses C_1 and C_2:*

$$\begin{aligned} C_1 &= son(X,Y) :- parent(Y,X). \\ C_2 &= son(jack,mary) :- male(jack), parent(mary,jack). \end{aligned}$$

The set representations of C_1 and C_2 are:

$$\begin{aligned} C_1 &= \{son(X,Y), \neg parent(Y,X)\}. \\ C_2 &= \{son(jack,mary), \neg male(jack), \neg parent(mary,jack)\} \end{aligned}$$

When $\theta = \{X/jack, Y/mary\}$,

$$\begin{aligned} C_1\theta &= \{son(jack,mary), \neg parent(mary,jack)\} \\ &\subseteq C_2 \end{aligned}$$

Thus, $C_1 \leq C_2$.

Definition 31 (Equivalence Classes of Clauses) *Let C and D be clauses. We say $C \sim D$ iff $C \leq D$ and $D \leq C$. Let $[C]$ denote the equivalence class under $\sim$ of C.*

Definition 32 (Ordering of Equivalence Classes) *Let C and D be clauses. We say that $[C] \leq [D]$ iff $C \leq D$.*

The set of equivalence classes forms a lattice. For further discussions, see Plotkin [43].

Definition 33 (Least General Generalization(LGG)) *Let M and N be terms, literals or clauses. L is a least general generalization of M and N iff*

(1) $L \leq M$ *and* $L \leq N$.

(2) If $L' \leq M$ *and* $L' \leq N$, *then* $L' \leq L$.

A least general generalization of M and N is the least upper bound of M and N in a lattice on the equivalence classes of clauses.

Definition 34 (Computation of an Lgg of Terms) *Given a signature* $\Sigma = \langle \mathcal{F}, \mathcal{P}, \mathcal{C} \rangle$ *and a set of variables* $\mathcal{V}$, *let* s *and* t *be terms. An operational definition of a function* $lgg(s,t)$ *that computes a least general generalization of* s *and* t *is defined as follows.*

1. $lgg(s,s) = s$ *if* s *is a constant in* $\mathcal{C}$.
2. $lgg(X,X) = X$ *if* X *is a variable in* $\mathcal{V}$.
3. $lgg(f^{(n)}(s_1,...,s_n), f^{(n)}(t_1,...,t_n)) = f^{(n)}(lgg(s_1,t_1),...,lgg(s_n,t_n))$.
4. *Otherwise,* $lgg(s,t) = V$, *where* V *is a new variable in* $\mathcal{V}$.

Definition 35 (Lgg of Atoms) *Given a signature* $\Sigma = \langle \mathcal{F}, \mathcal{P}, \mathcal{C} \rangle$, *let* P *and* Q *be atoms. An operational definition of a function* $lgg(P,Q)$ *that computes a least general generalization of* P *and* Q *is as follows.*

1. *If* $P = p^{(n)}(s_1,...,s_n)$ *and* $Q = p^{(n)}(t_1,...,t_n)$,
$$lgg(P,Q) = p^{(n)}(lgg(s_1,t_1),...,lgg(s_n,t_n)).$$
2. *Otherwise,* $lgg(P,Q)$ *is undefined.*

Definition 36 (Lgg of Literals) *Let* P *and* Q *be atoms and* L_1 *and* L_2 *be literals. An lgg of literals is defined as follows [28].*

1. *If* L_1 *and* L_2 *are atoms, then* $lgg(L_1,L_2)$ *is the lgg of atoms.*
2. *If* L_1 *and* L_2 *are the form* $\neg P$ *and* $\neg Q$, *respectively, then* $lgg(L_1,L_2) = lgg(\neg P, \neg Q) = \neg lgg(P,Q)$.
3. *Otherwise,* $lgg(L_1,L_2)$ *is undefined.*

Definition 37 (Lgg of Clauses) *Let clauses* $C = \{L_1,...,L_n\}$ *and* $D = \{K_1,...,K_m\}$. *Then* $lgg(C,D) = \{\ lgg(L_i,K_j) \mid L_i \in C, K_j \in D$ *and* $lgg(L_i,K_j)$ *is not undefined.*$\}$.

2.2.2 Relative Least General Generalization

This section introduces Plotkin's *relative least general generalization (rlgg)* according to [44].

Definition 38 *Let Th be a set of unit clauses.*

$$\overline{Th} = \{\sim L \mid \{L\} \in Th\},$$

where $\sim$ is defined as $\sim L = \neg L$ if L is a positive literal and $\sim \neg L = L$ if $\neg L$ is a negative literal.

An ordering of clauses relative to a set of unit clauses is defined as follows on the basis of the definition of ordering $\leq$ in the previous section.

Definition 39 (Relative Ordering of Literals) *Let L and M be literals. Let Th be a set of unit clauses. $L \leq M$ (Th) is defined as $\{L\} \leq \{M\} \cup \overline{Th}$.*

We read $L \leq M$ (Th) as meaning L is more general than M, relative to Th.

Definition 40 (Relative Ordering of Clauses) *Let C and D be clauses and Th be a set of unit clauses. $C \leq D$ (Th) is defined as $C \leq D \cup \overline{Th}$.*

Example 3 *Let C=$\{son(X,Y), \neg parent(Y,X)\}$, D=$\{son(jack,mary)\}$, and $Th = \{\{male(jack)\}, \{parent(mary,jack)\}\}$. C is more general than D, relative to Th, because for $\theta = \{X/jack, Y/mary\}$,*

$$\begin{aligned} C\theta &= \{son(jack,mary), \neg parent(mary,jack)\}. \\ D \cup \overline{Th} &= \{son(jack,mary), \neg male(jack), \neg parent(mary,jack)\}. \end{aligned}$$

Therefore, $C\theta \subseteq D \cup \overline{Th}$. Thus, $C \leq D$ (Th).

Note that $C \leq D$ does not hold without Th because D has no literal with the predicate symbol *parent*.

Definition 41 (Equivalence Class) *Let L and M be two literals or two clauses and Th be a set of clauses. We write $L \sim M$ (Th) when $L \leq M$ (Th) and $M \leq L$ (Th).*

Definition 42 (Relative Least General Generalization (RLGG)) *Let C, D and E be clauses and Th be a set of unit clauses. E is a least general generalization of C and D relative to Th iff*

(1) $E \leq C\ (Th)$ and $E \leq D\ (Th)$.

(2) If $E' \leq C\ (Th)$ and $E' \leq D\ (Th)$, then $E' \leq L\ (Th)$.

2.3 ψ-terms

This section introduces ψ-terms on the basis of *Order-Sorted Feature (OSF)* formalism [4, 2].

2.3.1 Syntax

Definition 43 (OSF Signature) *An OSF Signature is given by*

$$\Sigma_{OSF} = \langle \mathcal{P}, \mathcal{S}, \preceq, \sqcap, \sqcup, \mathcal{L} \rangle, \ s.t.:$$

- *$\mathcal{P}$ is a set of predicate symbols;*
- *$\mathcal{S}$ is a set of sort symbols with the sorts $\top$ and $\bot$;*
- *$\preceq$ is a partial order on $\mathcal{S}$ such that $\top$ is the greatest and $\bot$ is the least element;*
- *$\langle \mathcal{S}, \preceq, \sqcap, \sqcup \rangle$ is a lattice, where $s \sqcap t$ is defined as the infimum (or glb) of s and t and $s \sqcup t$ is the supremum (or lub) of sorts s and t;*
- *$\mathcal{L}$ is a set of feature symbols.*

Definition 44 *For $s_1, s_2 \in \mathcal{S}$, $s_1 \prec s_2$ iff $s_1 \preceq s_2$ and $s_1 \neq s_2$.*

Definition 45 (Constants) *A set of constants $\mathcal{C}$ is a subset of $\mathcal{S}$ such that $\mathcal{C}$=$\{c \in \mathcal{S} \mid$ for all $t \in \mathcal{S}$ if $t \prec c$ then $t = \bot\}$.*

Let $\mathcal{V}$ be a countable infinite set of variables.

Definition 46 (OSF-terms) *Given* $\Sigma_{OSF} = \langle \mathcal{P}, \mathcal{S}, \preceq, \sqcap, \sqcup, \mathcal{L} \rangle$, *if* $s \in \mathcal{S}$, $l_1, ..., l_n \in \mathcal{L}$, $X \in \mathcal{V}$, $n \geq 0$, *and* $t_1, ..., t_n$ *are OSF-terms, an OSF-term has the form*

$$X : s(l_1 \Rightarrow t_1, ..., l_n \Rightarrow t_n).$$

Let $\psi = X : s(l_1 \Rightarrow t_1, ..., l_n \Rightarrow t_n)$. X is called the root variable of ψ and s is called the root sort of ψ.

For a lighter notation, hereafter we will omit variables that are not shared and the sort of a variable when it is $\top$.

Definition 47 (ψ-terms) *An OSF-term*

$$\psi = X : s(l_1 \Rightarrow \psi_1, ..., l_n \Rightarrow \psi_n)$$

is in normal form (and then called a ψ-term) if:

- *For any variables V_i in ψ, V_i is the root variable of at most one non-top ψ-term*
- *s is a nonbottom sort in $\mathcal{S}$;*
- *$l_1, ..., l_n$ are pairwise distinct feature symbols in $\mathcal{L}$;*
- *$\psi_1, ..., \psi_n$ are ψ-terms.*

OSF-terms can be normalized to ψ-terms by OSF clause normalization rules, which are given in Section 2.3.3, unless the results include sort $\bot$ [4].

Definition 48 (Untagged ψ-terms) *Let* $\psi = X : s(l_1 \Rightarrow \psi_1, ..., l_n \Rightarrow \psi_n)$. $s(l_1 \Rightarrow \psi_1, ..., l_n \Rightarrow \psi_n)$ *is called an untagged ψ-term.*

Definition 49 (Feature Projection) *Given a ψ-term* $t = X : f(l_1 \Rightarrow t_1, ..., l_n \Rightarrow t_n)$, *the l_i projection of t (written as $t.l_i$) is defined as* $t.l_i = t_i$.

The definitions of atoms, literals, clauses, Horn clauses, and definite clauses are the same as those in Section 2.1. If features are non-zero integers 1,...,n, then a ψ-term $X : s(1 \Rightarrow t_1, 2 \Rightarrow t_2, ..., n \Rightarrow t_n)$ can be abbreviated to $X : s(t_1, t_2, ..., t_n)$.

Definition 50 (Ground Literal based on ψ-terms) *A literal L is a ground literal if all sorts in L are constants.*

Definition 51 (τ-terms) *A τ-term is a restricted form of a ψ-term s.t. $X : s$ if:*

- *X is a variable in $\mathcal{V}$;*
- *s is a sort symbol in $\mathcal{S}$.*

2.3.2 Semantics

Definition 52 (OSF Algebras) *An OSF Algebra is a structure $\mathcal{A} = \langle D^{\mathcal{A}}, (p^{\mathcal{A}})_{p\in\mathcal{P}}, (s^{\mathcal{A}})_{s\in\mathcal{S}}, (l^{\mathcal{A}})_{l\in\mathcal{L}}\rangle$ s.t.:*

- *$D^{\mathcal{A}}$ is non-empty set, called a domain of $\mathcal{A}$;*
- *for each predicate symbol $p \in \mathcal{P}$, $p^{\mathcal{A}} \subseteq (D^{\mathcal{A}})^n$;*
- *for each sort symbol $s \in \mathcal{S}$, $s^{\mathcal{A}} \subseteq D^{\mathcal{A}}$; in particular, $\top^{\mathcal{A}} = D^{\mathcal{A}}$ and $\bot^{\mathcal{A}} = \emptyset$;*
- *$(s \sqcap s')^{\mathcal{A}} = s^{\mathcal{A}} \cap s'^{\mathcal{A}}$ for two sorts $s, s' \in \mathcal{S}$;*
- *$(s \sqcup s')^{\mathcal{A}} = s^{\mathcal{A}} \cup s'^{\mathcal{A}}$ for two sorts $s, s' \in \mathcal{S}$;*
- *for each feature symbol $l \in \mathcal{L}$, $l^{\mathcal{A}} : D^{\mathcal{A}} \rightarrow D^{\mathcal{A}}$.*

Definition 53 ($\mathcal{A}$-Valuation) *Given $\Sigma_{OSF} = (\mathcal{P}, \mathcal{S}, \preceq, \sqcap, \sqcup, \mathcal{L})$, an $\mathcal{A}$-valuation is a function $\alpha : \mathcal{V} \rightarrow D^{\mathcal{A}}$.*

Definition 54 (Interpretation) *An interpretation $\mathcal{I}^{\mathcal{A}}$ is defined as $\mathcal{I}^{\mathcal{A}} = \langle \mathcal{A}, \alpha\rangle$, where $\mathcal{A}$ is an OSF Algebra and α is a $\mathcal{A}$-valuation.*

Definition 55 (Term Denotation) *Let t be a ψ-term of the form*

$$t = X : s(l_1 \Rightarrow t_1, ..., l_n \Rightarrow t_n).$$

Given an interpretation $\mathcal{I}^{\mathcal{A}}$, the term denotation of t is given by

$$[\![t]\!]^{\mathcal{A},\alpha} = \{\alpha(X)\} \cap s^{\mathcal{A}} \cap \bigcap_{1\leq i\leq n} (l_i^{\mathcal{A}})^{-1}([\![t_i]\!]^{\mathcal{A},\alpha}).$$

$$[\![t]\!]^{\mathcal{A}} = \bigcup_{\alpha:\mathcal{V}\rightarrow D^{\mathcal{A}}} [\![t]\!]^{\mathcal{A},\alpha}.$$

Definition 56 (Satisfaction Relation) *Given a signature* $\Sigma_{OSF} = \langle \mathcal{P}, \mathcal{S}, \preceq, \sqcap, \sqcup, \mathcal{L} \rangle$ *and an interpretation* $\mathcal{I}^{\mathcal{A}}$, *the satisfaction relation* $\models_{\mathcal{I}^{\mathcal{A}}}$ *is defined as follows.*

- $\models_{\mathcal{I}^{\mathcal{A}}} p(t_1, ..., t_n)$ *iff for all elements* $d_1, ..., d_n \in D^{\mathcal{A}}$ *such that* $\langle d_1, ..., d_n \rangle \in [\![\langle t_1, ..., t_n \rangle]\!]^{\mathcal{A}}$, $\langle d_1, ..., d_n \rangle \in p^{\mathcal{A}}$. *The notation* $[\![\langle t_1, ..., t_n \rangle]\!]^{\mathcal{A}}$ *is an abbreviation of*

$$\bigcup_{\alpha: \mathcal{V} \to D^{\mathcal{A}}} [\![t_1]\!]^{\mathcal{A},\alpha} \times ... \times [\![t_n]\!]^{\mathcal{A},\alpha} .$$

- $\models_{\mathcal{I}^{\mathcal{A}}} \neg F$ *iff* $\models_{\mathcal{I}^{\mathcal{A}}} F$ *does not hold.*
- $\models_{\mathcal{I}^{\mathcal{A}}} \{L_1, ..., L_n\}$ *iff* $L_1, ..., L_n$ *are literals and for some* i, $\models_{\mathcal{I}^{\mathcal{A}}} L_i$.
- $\models_{\mathcal{I}^{\mathcal{A}}} \{C_1, ..., C_n\}$ *iff* $C_1, ..., C_n$ *are clauses and for all* i, $\models_{\mathcal{I}^{\mathcal{A}}} C_i$.

Definition 57 (Herbrand Interpretation Equivalent) *Let a Herbrand universe* $HU = \{X_i{:}c_i \mid c_i \in \mathcal{C}, X_i \text{ in } \mathcal{V}\}$. *Let a Herbrand base*

$$HB = \{p(t_1, ..., t_n) \mid p \in \mathcal{P} \text{ and } t_1, ..., t_n \in HU\}.$$

Given an interpretation $\mathcal{I}^{\mathcal{A}} = \langle \mathcal{A}, \alpha \rangle$, *where* α *is an* $\mathcal{A}$*-valuation* $\alpha : \mathcal{V} \to D^{\mathcal{A}}$, *an interpretation* $\mathcal{I}^{\mathcal{A}}$ *is a Herbrand interpretation equivalent iff the following holds:*

- $D^{\mathcal{A}}$ *is a Herbrand universe* HU.
- *For all sorts* $c_i \in \mathcal{C}$, $c_i^{\mathcal{A}} = \{X_i : c_i\}$.
- *For all features* $l \in \mathcal{L}$, $l^{\mathcal{A}} : HU \to HU$.

2.3.3 Deduction over ψ-terms

Definition 58 (Sorted Substitution) *Let a sorted substitution have the form* $\{X_1{:}s_1/Y_1{:}t_1, ..., X_n{:}s_n/Y_n{:}t_n\}$, *where* $X_1, ..., X_n$ *and* $Y_1, ..., Y_n$ *are variables in* $\mathcal{V}$, $s_1, ..., s_n$ *and* $t_1, ..., t_n$ *are sort symbols in* $\mathcal{S}$, *and* $\bot \prec t_i \preceq s_i$ *for every* i. *If* E *is a term, a literal, or a clause,* $E\theta$ *is a result of replacing all occurrences of* $X_i{:}s_i$ *by* $Y_i{:}t_i$ *simultaneously for every* i.

An alternative syntactic presentation of the information conveyed by OSF-terms can be translated into a constraint clause [4].

Definition 59 (OSF-Constraints) *An order-sorted feature constraint (OSF-constraint) is an atomic expression of either of the forms:*

- $X : s$
- $X \doteq Y$
- $X.l \doteq Y$

where X and Y are variables in $\mathcal{V}$, s is a sort in $\mathcal{S}$, and l is a feature in $\mathcal{L}$.

Definition 60 (OSF-clauses) *An order-sorted feature clause (OSF-clause) $\phi_1 \& ... \& \phi_n$ is a finite, possibly empty conjunction of OSF-constraints $\phi_1, ..., \phi_n (n \geq 0)$.*

We can associate an OSF-term with a corresponding OSF-clause. Let ψ be a ψ-term of the form

$$\psi = X : s(l_1 \Rightarrow \psi_1, ..., l_n \Rightarrow \psi_n).$$

An OSF-clause $\phi(\psi)$ corresponding to an OSF-term ψ has the following form:

$$\begin{array}{lllllll} \phi(\psi) = X : s & \& & X.l_1 \doteq X'_1 & \& & ... & \& & X.l_n \doteq X'_n \\ & \& & \phi(\psi_1) & \& & ... & \& & \phi(\psi_n), \end{array}$$

where $X, X'_1, ..., X'_n$ are the root variables of $\psi, \psi_1, ..., \psi_n$, respectively. We say $\phi(\psi)$ is dissolved from the OSF-term ψ.

On the other hand, an OSF-clause ϕ can be converted to an OSF-term $\psi(\phi)$ as follows: first complete it by adding as many $V{:}\top$ constraints as needed so that there is exactly one sort constraint for every occurrence of a variable V in a $X.l{=}V$ constraint, where X is a variable and l is a feature symbol; then covert it by the following ψ transform:

$$\psi(\phi) = X : s(l_1 \Rightarrow \psi(\phi(Y_1)), ..., l_n \Rightarrow \psi(\phi(Y_n)))$$

where X is a root variable of ϕ, ϕ contains $X : s$, and $X.l_1 \doteq Y_1$,...,$X.l_n \doteq Y_n$ are all other constraints in ϕ with an occurrence of the variable X on the left-hand side. $\phi(Y)$ denotes the maximal subclause of ϕ rooted by Y.

Definition 61 (Solved OSF-Constraint) *An OSF-clause ϕ is called solved if for every variable X, ϕ contains:*

- *at most one sort constraint of the from $X : s$, with $\bot \prec s$;*
- *at most one feature constraint of the form $X.l \doteq Y$ for each l;*
- *no equality constraint of the form $X \doteq Y$.*

Given ϕ in normal form, we will refer to its part in solved form as $Solved(\phi)$.

Sort Intersection:
(1) $\frac{\phi \& X:s \& X:s'}{\phi \& X:s \sqcap s'}$

Inconsistent Sort:
(2) $\frac{\phi \& X:\bot}{X:\bot}$

Variable Elimination:
(3) $\frac{\phi \& X \doteq X'}{\phi[X/X'] \& X \doteq X'}$ if $X \neq X'$ and $X \in Var(\phi)$

Feature Decomposition:
(4) $\frac{\phi \& X.l \doteq X' \& X.l \doteq X''}{\phi \& X.l \doteq X' \& X' \doteq X''}$

Figure 2.1: OSF Clause Normalization Rules

Theorem 2 *[4] The rules of Fig. 2.1 are solution-preserving, finite-terminating, and confluent (modulo variable renaming). Furthermore, they always result in a normal form that is either the inconsistent OSF clause or an OSF clause in solved form together with a conjunction of equality constraints.*

Note that $Var(\phi)$ is the set of variables occurring in an OSF-clause ϕ and $\phi[X/Y]$ stands for the OSF-clause obtained from ϕ after replacing all occurrences of Y by X.

Theorem 3 (ψ-term Unification) *Let ψ_1 and ψ_2 be two ψ-terms. Let ϕ be the normal form of the OSF-clause $\phi(\psi_1)\&\phi(\psi_2)\&X_1 \doteq X_2$, where X_1 and X_2 are root variables of ψ_1 and ψ_2, respectively. Then, ϕ is the inconsistent clause iff their glb with respect to $\leq$ is $\bot$. If ϕ is not the inconsistent clause, then their glb $\psi_1 \sqcap \psi_2$ is given by the normal OSF-terms $\psi(Solved(\phi))$.*

Definition 62 (Typing Constraints) *A typing constraint is $X \doteq \psi$, where X is a variable and ψ is an OSF-term. Such an expression has this interpretation: $\models_{\mathcal{I}^{\mathcal{A}}} X \doteq \psi$ iff $\alpha(X) \in [\![\phi]\!]^{\mathcal{A},\alpha}$.*

A *resolvent* is the result of applying a resolution rule to a goal clause or a resolvent. For simplicity of natation, we consider all predicate symbols $r \in \mathcal{P}$ to be monadic.

Definition 63 (LIFE Resolution Rule) *A resolvent over OSF-terms $\mathbf{R} \equiv (:- R, r(\psi))$ reduces in one resolution step, choosing the literal $r(\psi)$ and the (renamed) program clause $r(\psi_0) :- r_1(\psi_1), ..., r_m(\psi_m)$ non-deterministically, to the resolvent*

$$\mathbf{R}' \equiv (:- R, r_1(\psi_1), ..., r_m(\psi_m), X \doteq (\psi \sqcap \psi_0)),$$

where X is the root variable of ψ.

Definition 64 (Deduction Relation over OSF-terms) *Given sets of definite clauses T and D based on ψ-terms, the deduction relation $T \vdash_{OSF} D$ holds iff D is proven from T by the LIFE Resolution [2] in which the LIFE resolution rule is iteratively applied.*

Theorem 4 (Correctness of LIFE Resolution) *[2] The LIFE resolution for definite clauses and goal clause based on OSF-terms is complete and sound.*