The defining power of stratified and hierarchical logic programs

Gerhard Jäger and Robert F. Stärk

Abstract

We investigate the defining power of stratified and hierarchical logic programs. As an example for the treatment of negative information in the context of these structured programs we also introduce a stratified and hierarchical closed-world assumption. Our analysis tries to relate the defining power of stratified and hierarchical programs (with and without an appropriate closed-world assumption) very precisely to notions and hierarchies in classical definability theory.

Stratified and hierarchical logic programs are two well-known and typical candidates of what one may more generally denote as *structured* programs. In both cases we have to deal with normal logic programs which satisfy certain syntactic conditions with respect to the occurrence of negative literals. Recently they have gained a lot of importance in connection with the search for nice declarative semantics for logic programs and the treatment of negative information in logic programming (e.g., Lloyd [10]).

Stratified programs were introduced into logic programming by Apt, Blair, and Walker [2] and van Gelder [17] not long ago. In mathematical logic, however, theories of this kind have been studied for more than 20 years under the general theme of iterated inductive definability. Indeed, stratified programs can be understood as systems for (finitely) iterated inductive definitions where the definition clauses are of very low logical complexity. The notion of hierarchical program (e.g., Clark [6], Shepherdson [15]), on the other hand, is motivated by database theory and tries to reflect the idea of iterated explicit definability by simple principles.

From a conceptual point of view we are interested in the relationship between logic programming, inductive definability and equational definability. By making use of these connections we obtain a uniform and perspicuous approach to a series of interesting questions in this area.

Address correspondence to Gerhard Jäger or Robert F. Stärk, Institut für Informatik und angewandte Mathematik, Länggassstrasse 51, CH-3012 Bern, Switzerland.
Email: ⟨jaeger@iam.unibe.ch⟩ or ⟨staerk@iam.unibe.ch⟩

Appeared in: *J.of Logic Programming*, 15 (1&2): 55–77, 1993.

The plan of this paper is as follows: Section 1 introduces some basic notions. Sections 2 and 3 present the relevant concepts from classical definability theory and are concerned with various forms of definability over Herbrand universes of first-order languages. In Section 4 we characterize the defining power of stratified programs. Among other things we prove that the arithmetically definable subsets of the non-negative integers comprise the defining power of suitable stratified programs with the stratified closed-world assumption. Section 5 is then devoted to the study of hierarchical programs. It is shown that definite hierarchical programs pin down exactly the so-called *term-definable* relations. This is in sharp contrast to the defining power of arbitrary hierarchical logic programs which is shown to be equivalent to that of definite programs. Finally the hierarchical programs with the hierarchical closed-world assumption represent a class of intermediate strength. We will see that they exactly define the *equationally definable* relations.

1 Basic notions

First we have to introduce some basic terminology and definitions. We will try to follow the standard terminology of logic programming as far as possible and use Lloyd [10] as standard reference for unexplained notions and results.

We start out from countable first-order languages L with equality which satisfy the following conditions with respect to their function and relation symbols:

(1) L contains a finite number of function symbols;

(2) L contains at least one 0-ary function symbol;

(3) L contains countably many relation symbols $P, Q, R, P_1, Q_1, R_1, \ldots$ of every finite arity.

First-order languages of this kind are called *finite* languages by Shepherdson [16]. In the context of logic programming the restriction to finitely many function symbols seems justified since every logic program only involves a finite number of function symbols. Observe, however, that logic programming is very sensitive with respect to the function symbols of the underlying language. In general the meaning $m(T, L)$ of a logic program T with respect to the language L is different from $m(T, L_f)$ if L_f is the extension of L by a new function symbol f. On the other hand extensions of languages by additional relation symbols are completely unproblematic, and we have $m(T, L) = m(T, L_R)$ for all extensions L_R of L by a new relation symbol R. Therefore, we are free to assume that the underlying language contains an arbitrary number of relation symbols.

The terms $s, t, s_1, t_1, \ldots$ and formulas $\varphi, \psi, \chi, \theta, \varphi_1, \psi_1, \chi_1, \theta_1, \ldots$ of L are defined as usual; terms and formulas without free variables are called *ground*, 0-ary function symbols are called *constants*. Hence condition (2) guarantees the existence of a ground L term. The *literals* $F, G, F_1, G_1, \ldots$ of L are the atomic formulas and negated atomic formulas of L. Relation symbols different from the equality symbol are denoted as *proper* relation symbols; proper literals (proper atomic formulas) are literals (atomic formulas) which do not contain the equality symbol.

As usual, the *Herbrand universe* U_L denotes the collection of all ground terms of L and the *Herbrand base* B_L the collection of all ground atomic formulas of L. An L *theory* is a (possibly infinite) collection of L formulas. By $T \vdash \varphi$, we express that the formula φ can be deduced from the theory T by the usual axioms and rules of predicate logic with equality. Finally, a *normal clause* in L is an L formula φ of the form

$$F_1 \wedge \ldots \wedge F_n \ \rightarrow \ G$$

with $n \geq 0$, where G is a proper atomic formula and $F_1, \ldots, F_n$ are proper literals; φ is called *definite* if also the $F_1, \ldots, F_n$ are atomic. A *normal program* in L is a finite set of normal clauses in L, and a *definite program* in L is a finite collection of definite clauses in L.

The vector notation $\vec{V}$ is used as shorthand for a finite string $V_1, \ldots, V_n$ whose length will be specified by the context. We write $\varphi[\vec{R}, \vec{x}]$ to indicate that all proper relation symbols of the formula φ come from the list $\vec{R}$ and all free variables of φ from the list $\vec{x}$; analogously, $t[\vec{x}]$ stands for a term with no variables different from $\vec{x}$. The formula $\varphi(\vec{R}, \vec{x})$ and the term $t(\vec{x})$ may contain other relation symbols and free variables besides $\vec{R}$ and $\vec{x}$. In addition, if $\vec{A} = A_1, \ldots, A_n$, then the notation $\vec{A} \subset U_L$ is supposed to express that $A_1, \ldots, A_n$ are (arbitrary) relations on U_L; it does not imply, however, that $A_1, \ldots, A_n$ are subsets of U_L.

Now choose an L formula $\varphi[\vec{R}, \vec{x}]$, relations $\vec{A} \subset U_L$ and elements $\vec{a} \in U_L$. Then by $U_L \models \varphi[\vec{A}, \vec{a}]$, we mean that φ is valid in the Herbrand structure with universe U_L, provided that the relation symbols $\vec{R}$ are interpreted by the relations $\vec{A}$ and the free variables $\vec{x}$ by the elements $\vec{a}$.

We write $\sim A$ for the complement $\{\langle \vec{a} \rangle \in U_L^n : \langle \vec{a} \rangle \notin A\}$ of an n-ary relation A on U_L. If A is a subset of U_L^{m+1}, then each subset of U_L^m of the form

$$\{\langle a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_{m+1} \rangle \in U_L^m : \langle a_1, \ldots, a_{i-1}, b, a_{i+1}, \ldots, a_{m+1} \rangle \in A\}$$

for some $b \in U_L$ is called a *section* of A. If f is an n-ary function symbol of L, then the *graph* $Gr(f)$ of f is the $(n+1)$-ary relation on U_L defined as

$$Gr(f) := \{\langle \vec{a}, f(\vec{a}) \rangle : \vec{a} \in U_L\}.$$

If $\mathcal{C}$ is a collection of L formulas, $\mathcal{K}$ a collection of relations on U_L and $B \subset U_L^n$, then B is called $\mathcal{C}$ *definable in* L *with parameters from* $\mathcal{K}$ if there exists a formula $\varphi[\vec{R}, \vec{x}]$ in $\mathcal{C}$ and a sequence $\vec{A}$ of elements of $\mathcal{K}$ of appropriate arities such that

$$\langle \vec{a} \rangle \in B \iff U_L \models \varphi[\vec{A}, \vec{a}]$$

for all $\vec{a} \in U_L$. The class of all relations which are $\mathcal{C}$ definable in L with parameters from $\mathcal{K}$ is denoted by $\mathcal{C}(\mathcal{K}, L)$. B is called $\mathcal{C}$ *definable in* L if it belongs to $\mathcal{C}(\emptyset, L)$.

An n-ary relation A on U_L is called *definable* by the L theory T if there exists an n-ary relation symbol R of L so that we have

$$\langle \vec{a} \rangle \in A \iff T \vdash R(\vec{a})$$

for all $\vec{a} \in U_L$. The collection of all T definable relations on U_L is then denoted by $Def_L(T)$.

Later we will also make some remarks about definability over the non-negative integers $\mathbb{N}$. To fit this concept into our present framework, we fix a finite first-order language L_N with exactly one constant 0 and one unary function symbol S_u. This function symbol represents a successor function, and the Herbrand universe U_{L_N} of L_N, which we simply denote as U_N, may be regarded as an isomorphic copy of $\mathbb{N}$, where the L_N term $S_u^n(0)$ corresponds to the natural number n,

$$U_N = \{0, S_u(0), S_u(S_u(0)), \ldots\}.$$

2 Term and equationally definable relations

In this and the next section we introduce the tools from definability theory which will be used later in order to characterize the defining power of stratified and hierarchical programs with and without suitable forms of the closed-world assumption. We focus on three definition principles: (1) term definability, i.e., explicit definability by means of terms of the language; (2) explicit definability by equational formulas; and (3) inductive definability by positive Σ formulas.

Definition 2.1. Let A be a subset of U_L^n.

(1) A is called *locally term-definable in* L if there exist terms $t_1[\vec{x}], \ldots, t_n[\vec{x}]$ so that

$$A = \{\langle t_1[\vec{a}], \ldots, t_n[\vec{a}] \rangle : \vec{a} \in U_L\}.$$

(2) A is called *term-definable in* L if A is a finite union of locally term-definable relations.

From this definition we immediately obtain that the empty set $\emptyset$ is term-definable and that the finite union of term-definable sets is term-definable. The closure of term-definable relations under intersection requires some (easy) arguments.

Lemma 2.1. *The intersection of finitely many term-definable subsets of U_L^n is term-definable.*

PROOF. It is sufficient to show that the intersection of two locally term-definable sets is term-definable. Hence let

$$A := \{\langle s_1[\vec{a}], \ldots, s_n[\vec{a}]\rangle : \vec{a} \in U_L\} \quad \text{and} \quad B := \{\langle t_1[\vec{a}], \ldots, t_n[\vec{a}]\rangle : \vec{a} \in U_L\}$$

be two locally term-definable subsets of U_L^n. We may assume that the variables of $s_1, \ldots, s_n$ do not occur in $t_1, \ldots, t_n$. If $A \cap B = \emptyset$, then $A \cap B$ is term-definable; otherwise, there exists a most general unifier σ of $\langle s_1, \ldots, s_n\rangle$ and $\langle t_1, \ldots, t_n\rangle$, and obviously $A \cap B = \{\langle s_1\sigma[\vec{a}], \ldots, s_n\sigma[\vec{a}]\rangle : \vec{a} \in U_L\}$. □

Lemma 2.2. *If $A \subset U_L^n$ is term-definable, then for every $1 \leq i \leq n$,*

$$\{\langle a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n\rangle : (\exists b \in U_L)(\langle a_1, \ldots, a_{i-1}, b, a_{i+1}, \ldots, a_n\rangle \in A)\}$$

is a term-definable subset of U_L^{n-1}. Hence the collection of term-definable relations in L is closed under projections.

Example 2.1.

(1) The unary relations on U_N which are term-definable in L_N are exactly the $A \subset U_N$ so that A or $\sim A$ is finite.

(2) The set $B := \{\langle a, a\rangle : a \in U_N\}$ is term-definable, but $\sim B$ is not term-definable.

Term-definable relations will be important for describing the defining power of definite logic programs and are closely related to the parameter-free Σ_1^+ relations introduced below (cf. Lemma 2.7). Now we turn to a more general notion and call an L formula an *equational* formula of L if it does not contain proper relation symbols. It follows from the previous definitions that every relation A on U_L which is term-definable in L is also equationally definable in L. The converse is not correct, as one can easily see by the following example: the relation $\{\langle a, b\rangle \in U_N^2 : a \neq b\}$ is equationally definable but not term-definable.

Shepherdson's article [16] is devoted to the equality theory in the context of logic programming. Besides many other results it proves the following reduction property, which will help us later in comparing the strength of hierarchical programs with and without the hierarchical closed-world assumption.

Lemma 2.3. (Reduction property of equational formulas). For every equational L formula $\varphi[\vec{x}]$ there exist finitely many strictly simple equality formulas $\psi_1[\vec{x}]$, ..., $\psi_n[\vec{x}]$ of L, so that we have for all $\vec{a} \in U_L$:

$$U_L \models \varphi[\vec{a}] \leftrightarrow \psi_1[\vec{a}] \vee \ldots \vee \psi_n[\vec{a}].$$

Here, an L formula is called strictly simple *if it is of the form*

$$(\exists \vec{y})\Big(\bigwedge_{i \in I} x_i = r_i[\vec{x}, \vec{y}] \wedge \bigwedge_{j \in J} x_{\sigma(j)} \neq s_j[\vec{x}, \vec{y}] \wedge \bigwedge_{k \in K} y_{\tau(k)} \neq t_k[\vec{x}, \vec{y}] \Big),$$

where $\vec{x} = x_1, \ldots, x_m$ and $\vec{y} = y_1, \ldots, y_l$ and

- *$I \subset \{1, \ldots, m\}$;*
- *$\{\sigma(j) : j \in J\} \subset \{1, \ldots, m\} \setminus I$ and $\{\tau(k) : k \in K\} \subset \{1, \ldots, l\}$;*
- *each x_i for $i \in I$ does not occur anywhere in the formula except on the left-hand side of $x_i = r_i[\vec{x}, \vec{y}]$;*
- *each y_j of $\vec{y}$ occurs in one of the terms $r_i[\vec{x}, \vec{y}]$ for some $i \in I$.*

Applied to the special case $L = L_N$, this lemma has the consequence that a subset $A \subset U_N$ is equationally definable if and only if A or $\sim A$ is finite. Hence term definability in L_N is equivalent to equational definability in L_N as far as unary relations are concerned. However, as we have seen above, this equivalence cannot be extended to, for example, binary relations.

The class Σ^+ of *positive existential L* formulas is inductively defined as follows:

(1) If s and t are terms, then $(s = t)$ belongs to Σ^+.

(2) If R is an n-ary relation symbol of L and and $t_1, \ldots, t_n$ are terms, then the formula $R(t_1, \ldots, t_n)$ belongs to Σ^+.

(3) If φ and ψ belong to Σ^+, then so do $(\varphi \vee \psi)$ and $(\varphi \wedge \psi)$.

(4) If $\varphi(x)$ belongs to Σ^+, then so does $(\exists x)\varphi(x)$.

A formula φ is a Σ_1^+ *formula* of L if it is a Σ^+ formula of the form

$$(\exists x_1) \ldots (\exists x_k)\psi(x_1, \ldots, x_k),$$

where ψ does not contain quantifiers. The class $r\Sigma_1^+$ of the *relational* Σ_1^+ formulas of L consists of all Σ_1^+ formulas of L without function symbols.

It is obvious that every Σ^+ formula of L is equivalent to a Σ_1^+ formula of L with the same relation and function symbols and the same free variables. A reduction of the Σ^+ to the $r\Sigma_1^+$ is possible and described in the following lemma. Its proof is based on the usual representation of functions by their graphs and will be omitted.

Lemma 2.4. Let $\varphi[\vec{R},\vec{x}]$ be a Σ^+ formula of L with no function symbols different from $f_1,\ldots,f_m$. Then, there exists a $r\Sigma_1^+$ formula $\psi[\vec{Q},\vec{R},\vec{x}]$ of L so that we have for all $\vec{A}\subset U_L$ and $\vec{a}\in U_L$:

$$U_L \models \varphi[\vec{A},\vec{a}] \iff U_L \models \psi[Gr(f_1),\ldots,Gr(f_m),\vec{A},\vec{a}].$$

Σ^+ formulas have very simple normal forms with respect to the equality symbol and the other relation symbols which they contain. These normal forms also will provide a convenient tool for reducing Σ^+ inductively definable sets to suitable logic programs.

Definition 2.2. An L formula $\varphi[\vec{R},x_1,\ldots,x_m]$ is called *molecular* if it is of the form

$$(\exists\vec{y})(\ x_1 = t_1[\vec{y}] \wedge \ldots \wedge x_m = t_m[\vec{y}] \wedge F_1[\vec{y}] \wedge \ldots \wedge F_n[\vec{y}]\ ),$$

where the variables $x_1,\ldots,x_m$ and $\vec{y}$ are pairwise different and the $F_i[\vec{y}]$ are positive literals but no equations.[1]

Lemma 2.5. (Normal form of Σ^+ formulas). Let $\varphi[\vec{R},\vec{x}]$ be a Σ^+ formula of L. Then, there exist finitely many molecular L formulas $\psi_1[\vec{R},\vec{x}],\ldots,\psi_n[\vec{R},\vec{x}]$, so that we have for all $\vec{A}\subset U_L$ and all $\vec{a}\in U_L$:

$$U_L \models \varphi[\vec{A},\vec{a}] \leftrightarrow \psi_1[\vec{A},\vec{a}] \vee \ldots \vee \psi_n[\vec{A},\vec{a}].$$

Proof. First we replace $\varphi[\vec{R},\vec{x}]$ by a logically equivalent formula

$$\chi_1[\vec{R},\vec{x}] \vee \ldots \vee \chi_r[\vec{R},\vec{x}],$$

where every $\chi_i[\vec{R},\vec{x}]$ is of the form

$$(\exists\vec{y})(r_1[\vec{x},\vec{y}] = s_1[\vec{x},\vec{y}] \wedge \ldots \wedge r_m[\vec{x},\vec{y}] = s_m[\vec{x},\vec{y}] \wedge F_1[\vec{x},\vec{y}] \wedge \ldots \wedge F_l[\vec{x},\vec{y}]),$$

$\vec{x} = x_1,\ldots,x_g$ and $\vec{y} = y_1,\ldots,y_h$. Each $\chi_i[\vec{R},\vec{x}]$ will now be transformed into a molecular formula, so that every transformation step is valid in U_L. To achieve this, we put

$$r_1[\vec{x},\vec{y}] = s_1[\vec{x},\vec{y}] \wedge \ldots \wedge r_m[\vec{x},\vec{y}] = s_m[\vec{x},\vec{y}]$$

by a version of the unification algorithm into the solved form

$$\bigwedge_{i\in I} x_i = t_i[\vec{x},\vec{y}] \wedge \bigwedge_{j\in J} y_j = t_j^*[\vec{x},\vec{y}].$$

[1] Hence a molecular formula of this form contains no free variables different from $x_1,\ldots,x_m$.

In this expression I is a subset of $\{1,\ldots,g\}$, J a subset of $\{1,\ldots,h\}$ and the variables x_i for $i \in I$ and y_j for $j \in J$ do not occur in any of the terms on the right-hand side of the equations. Now we define

$$\chi_i'[\vec{R},\vec{x}] := (\exists \vec{y})\Big(\bigwedge\!\!\!\!\bigwedge_{i\in I} x_i = t_i[\vec{x},\vec{y}] \wedge F_1'[\vec{x},\vec{y}] \wedge \ldots \wedge F_l'[\vec{x},\vec{y}]\Big)$$

with $F_k'[\vec{x},\vec{y}]$ denoting the atomic formula which results from $F_k[\vec{x},\vec{y}]$ by replacing x_i by $t_i[\vec{x},\vec{y}]$ for $i \in I$ and y_j by $t_j^*[\vec{x},\vec{y}]$ for $j \in J$. Observe that the variables x_i for $i \in I$ and y_j for $j \in J$ do not occur in $F_1'[\vec{x},\vec{y}] \wedge \ldots \wedge F_l'[\vec{x},\vec{y}]$. Now define

$$\psi_i[\vec{R},\vec{x}] := (\exists \vec{v})(\exists \vec{y})\Big(\bigwedge\!\!\!\!\bigwedge_{i\in I} x_i = t_i[\vec{x},\vec{y}] \wedge \bigwedge\!\!\!\!\bigwedge_{k\in K} x_k = v_k \wedge F_1'[\vec{v},\vec{y}] \wedge \ldots \wedge F_l'[\vec{v},\vec{y}]\Big)$$

for $\vec{v} = v_1,\ldots,v_g$ and $K := \{1,\ldots,g\} \setminus I$. The formula $\psi_i[\vec{R},\vec{x}]$ is the desired transformation of $\chi_i[\vec{R},\vec{x}]$ in molecular form. □

If $\mathcal{K}$ is a collection of relations on U_L, then $\Sigma_1^+(\mathcal{K},L)$ has been defined to be the class of all relations on U_L that are Σ_1^+ definable in L with parameters from $\mathcal{K}$. Some of the closure properties of this class are listed in the following lemma.

Lemma 2.6. Let $\mathcal{K}$ be a collection of relations on U_L.

(1) $\Sigma_1^+(\mathcal{K},L)$ is closed under finite unions, finite intersections and sections.

(2) $\Sigma_1^+(\Sigma_1^+(\mathcal{K},L),L) = \Sigma_1^+(\mathcal{K},L)$.

The proofs of these assertions are immediate from the definition of Σ_1^+ definability. The class $\Sigma_1^+(\emptyset,L)$ is of special interest since it corresponds to the collection of term-definable relations in L.

Lemma 2.7. The class $\Sigma_1^+(\emptyset,L)$ consists exactly of the relations on U_L which are term-definable in L.

Proof. It is obvious that every relation A on U_L which is locally term-definable in L belongs to $\Sigma_1^+(\emptyset,L)$. In view of the previous lemma, one can therefore conclude that $\Sigma_1^+(\emptyset,L)$ contains the term-definable relations on U_L. Now suppose that the relation $A \subset U_L^n$ is defined by the Σ_1^+ formula $\varphi[x_1,\ldots,x_n]$. By Lemma 2.5 the normal form of $\varphi[x_1,\ldots,x_n]$ is

$$\bigvee_{i=1}^{m} (\exists \vec{y})(x_1 = t_{i,1}[\vec{y}] \wedge \ldots \wedge x_n = t_{i,n}[\vec{y}]),$$

so that $A = \bigcup_{i=1}^m \{\langle t_{i,1}[\vec{a}],\ldots,t_{i,n}[\vec{a}]\rangle : \vec{a} \in U_L\}$. Hence, A is term-definable in L. □

The following lemma will be used in Section 4. Its proof is straightforward by induction on the complexity of the Σ^+ formula involved.

Lemma 2.8. Let T by an arbitrary L theory, $\varphi[R_1, \ldots, R_m, \vec{x}]$ a Σ^+ formula of L and $\vec{A} = A_1, \ldots, A_m$ a sequence of relations on U_L which satisfy

$$\langle \vec{a} \rangle \in A_i \quad \Longrightarrow \quad T \vdash R_i(\vec{a})$$

for all $1 \leq i \leq m$ and all $\vec{a} \in U_L$. Then we have for all $\vec{b} \in U_L$:

$$U_L \models \varphi[\vec{A}, \vec{b}] \quad \Longrightarrow \quad T \vdash \varphi[\vec{R}, \vec{b}].$$

3 Inductively definable relations

In order to characterize the defining power of stratified logic programs, we will make use of some concepts from the theory of inductive definitions as it is developed for example in the textbooks by Barwise [5], Hinman [8] and Moschovakis [13]. Hence, suppose that $\vec{R} = R_1, \ldots, R_k$, $\vec{x} = x_1, \ldots, x_m$ and that $\varphi[Q, \vec{R}, \vec{x}]$ is a Σ^+ formula of L. In addition we assume that $\vec{A} = A_1, \ldots, A_k$ is a sequence of relations on U_L.

Then we define by recursion on the ordinals the following subsets of U_L^m:

$$\begin{aligned} I_\varphi^{<\alpha}(\vec{A}) &:= \bigcup_{\xi<\alpha} I_\varphi^\xi(\vec{A}) \,, \\ I_\varphi^\alpha(\vec{A}) &:= \{\langle \vec{a} \rangle \in U_L^m : U_L \models \varphi[I_\varphi^{<\alpha}(\vec{A}), \vec{A}, \vec{a}]\} \\ I_\varphi(\vec{A}) &:= \bigcup_{\xi \in On} I_\varphi^\xi(\vec{A}) \,. \end{aligned}$$

Inductive definitions are studied at full length in the literature. In our special case we can conclude, for example, that (with the assumptions mentioned above)

(1) there exists an ordinal $\alpha \leq \omega$ so that

$$I_\varphi^{<\alpha}(\vec{A}) = I_\varphi^\alpha(\vec{A}) = I_\varphi(\vec{A});$$

(2) $I_\varphi(\vec{A})$ is the least fixed point of the operator $\Gamma_{\varphi,\vec{A}}$ which is defined for all $X \subset U_L^m$ by

$$\Gamma_{\varphi,\vec{A}}(X) := \{\langle \vec{a} \rangle \in U_L^m : U_L \models \varphi[X, \vec{A}, \vec{a}]\}.$$

In the following, we will be interested in relations on U_L which can be defined inductively over the Herbrand universe U_L.

Definition 3.1. Let $\mathcal{K}$ be a collection of relations on U_L.

(1) The class Σ^+-$FP(\mathcal{K}, L)$ of the Σ^+ *fixed points with parameters from* $\mathcal{K}$ consists of all relations on U_L of the form $I_\varphi(\vec{A})$ such that $\vec{A} \in \mathcal{K}$ and $\varphi[Q, \vec{R}, \vec{x}]$ is a Σ^+ formula of L of the appropriate arities.

(2) The class Σ^+-$IND(\mathcal{K}, L)$ of the Σ^+ *inductively defined sets with parameters from* $\mathcal{K}$ is the least collection of relations on U_L which contains Σ^+-$FP(\mathcal{K}, L)$ and is closed under sections.

Moschovakis [13] contains a series of results concerning the closure properties of classes of inductively defined sets, for example, the *simultaneous induction lemma*, the *combination lemma* and the *transitivity theorem*. Applied to our context, we have the following basic properties of the classes Σ^+-$IND(\mathcal{K}, L)$.

Remark 3.1. Let $\mathcal{K}$ be a collection of relations on U_L.

(1) $\Sigma^+_1(\mathcal{K}, L) \subset \Sigma^+$-$IND(\mathcal{K}, L)$.

(2) Σ^+-$IND(\Sigma^+$-$IND(\mathcal{K},), L) = \Sigma^+$-$IND(\mathcal{K}, L)$.

(3) In general, the classes Σ^+-$IND(\mathcal{K}, L)$ will not be closed under complements.

Motivated by this observation, we now introduce the hierarchy $\langle \mathcal{EI}_n(L) : n < \omega \rangle$ of *iterated existential inductive relations* on U_L. If $\mathcal{K}$ is a class of relations on U_L, then we write $\overline{\mathcal{K}}$ for the collection of their complements, i.e.,

$$\overline{\mathcal{K}} := \{\sim A : A \in \mathcal{K}\}.$$

Definition 3.2. By induction on the natural numbers n, we define the classes $\mathcal{EI}_n(L)$ of relations on U_L:

$$\begin{aligned} \mathcal{EI}_0(L) &= \Sigma^+\text{-}IND(\emptyset, L), \\ \mathcal{EI}_{n+1}(L) &= \Sigma^+\text{-}IND(\mathcal{EI}_n(L) \cup \overline{\mathcal{EI}_n(L)}, L). \end{aligned}$$

For notational simplicity, we will write $\mathcal{EI}_n(N)$ instead of $\mathcal{EI}_n(L_N)$. The following observation is obvious.

Lemma 3.1.

(1) $\mathcal{EI}_n(L) \subset \mathcal{EI}_{n+1}(L)$ *for all natural numbers* n.

(2) The graph $Gr(f)$ *of a function symbol* f *of* L *is an element of* $\mathcal{EI}_0(L)$.

In view of the reduction property of equational formulas we know that all equationally definable (in L) relations are contained in $\mathcal{EI}_0(L) \cap \overline{\mathcal{EI}_0(L)}$. In general, however, there will be elements of $\mathcal{EI}_0(L)$ which are not equationally definable in L. A typical example is the subset $\{s_u^{2n}(0) : n \in \mathbb{N}\}$ of U_N which is inductively but not equationally definable.

The next theorem summarizes some basic properties of inductive definability over the natural numbers $\mathbb{N}$ in terms of the classes $\mathcal{EI}_n(N)$. For the proof of this theorem we refer to the respective section in Hinman [8].

Theorem 3.1. *We have the following for all natural numbers n:*

(1) *$\mathcal{EI}_0(N)$ is the class of the recursively enumerable subsets of the natural numbers $\mathbb{N}$, i.e., the class of the Σ_1 subsets of $\mathbb{N}$.*

(2) *If $\mathcal{K}$ is a collection of relations on U_N which contains $\mathcal{EI}_0(N)$ and is closed under complements, then we have: Σ^+-$IND(\mathcal{K}, L_N) = \Sigma_1^+(\mathcal{K}, L_N)$.*

(3) *$\mathcal{EI}_{n+1}(N) = \Sigma_1^+(\mathcal{EI}_n(N) \cup \overline{\mathcal{EI}_n(N)}, L_N)$, hence $\mathcal{EI}_{n+1}(N)$ is the class of the Σ_{n+2} subsets of the natural numbers $\mathbb{N}$.*

Next, we turn to the relationship between definability over U_L and U_N. Similar observations have been made by various authors, such as Andreka and Nemeti [1] and Apt [3]. However, the approach presented here is more closely tied to the notion of inductive definability.

In a first step, we reduce fixed points of Σ^+ formulas to those of $r\Sigma_1^+$ formulas. As a consequence of Lemma 2.4, we obtain:

Corollary 3.1. *Let $\varphi[P, \vec{Q}, \vec{x}]$ be a Σ^+ formula of L with no function symbols different from $f_1, \ldots, f_m$. Then, there exists a $r\Sigma_1^+$ formula $\psi[P, \vec{Q}, \vec{R}, \vec{x}]$ of L so that we have for all $\vec{A} \subset U_L$:*

$$I_\varphi(\vec{A}) = I_\psi(\vec{A}, Gr(f_1), \ldots, Gr(f_m)).$$

Until the end of this section, we assume that L is a language with finitely many function symbols and that at least one of this function symbols has an arity greater than 0. Then, there exists a mapping β from U_L to U_N,

$$\beta: U_L \to U_N,$$

which is one to one and onto. If A is a subset of U_L^n, then we define

$$\beta(A) := \{\langle \beta(a_1), \ldots, \beta(a_n)\rangle : \langle a_1, \ldots, a_n\rangle \in A\}$$

and write $\beta(\vec{b})$ and $\beta(\vec{B})$ instead of $\beta(b_1), \ldots, \beta(b_m)$ and $\beta(B_1), \ldots, \beta(B_n)$ for all $\vec{b} = b_1, \ldots, b_m \in U_L$ and $\vec{B} = B_1, \ldots, B_n \subset U_L$, respectively.

An obvious induction on the length of the $r\Sigma_1^+$ formulas $\varphi[\vec{R},\vec{x}]$ of L then yields

$$U_L \models \varphi[\vec{A},\vec{a}] \iff U_N \models \varphi[\beta(\vec{A}),\beta(\vec{a})]$$

for all $\vec{a} \in U_L$ and $\vec{A} \subset U_L$. Using this observation, it is easy to show that we have for all ordinals α, $\vec{a} \in U_L$, $\vec{A} \subset U_L$ and all $r\Sigma_1^+$ formulas $\varphi[Q,\vec{R},\vec{x}]$

$$\langle\vec{a}\rangle \in I_\varphi^\alpha(\vec{A}) \iff \langle\beta(\vec{a})\rangle \in I_\varphi^\alpha(\beta(\vec{A})),$$

where the sets $I_\varphi^\alpha(\vec{A})$ are defined over U_L and the sets $I_\varphi^\alpha(\beta(\vec{A}))$ over U_N. As a consequence, we obtain the following isomorphism between the sets $I_\varphi(\vec{A})$ defined over U_L and the sets $I_\varphi(\beta(\vec{A}))$ defined over U_N:

$$\beta(I_\varphi(\vec{A})) = I_\varphi(\beta(\vec{A})),$$

provided that $\varphi[Q,\vec{R},\vec{x}]$ is a $r\Sigma_1^+$ formula of L.

It is straightforward but tedious to show that we can choose the bijection β so that the following two conditions are satisfied:

(β.1) If f is a function symbol of L, then $\beta(Gr(f)) \in \mathcal{EI}_0(N)$.

(β.2) There exist $D_0, D_1 \in \mathcal{EI}_0(L)$ so that $\beta(D_0) = Gr(0)$ and $\beta(D_1) = Gr(S_u)$.

This correspondence plays an important role in the proof of the following theorem.

Theorem 3.2. *Let L and β be described as above. Then we have for all natural numbers n:*

(1) *If $A \in \mathcal{EI}_n(L)$, then $\beta(A) \in \mathcal{EI}_n(N)$.*

(2) *If $B \in \mathcal{EI}_n(N)$, then there exists an $A \in \mathcal{EI}_n(L)$, so that $B = \beta(A)$.*

Proof. We prove both assertions simultaneously by complete induction on n.

I. $n = 0$. It is sufficient for the first assertion to show that $\beta(A) \in \mathcal{EI}_0(N)$ for all $A \in \Sigma^+$-$FP(\emptyset, L)$. So assume that we have a Σ^+ formula $\varphi[P,\vec{x}]$ of L with no function symbols different from $f_1,\dots,f_m$ such that $A = I_\varphi(-)$. By Corollary 3.1, there exists an $r\Sigma_1^+$ formula $\psi[P,\vec{Q},\vec{x}]$ with the property

$$A = I_\psi(Gr(f_1),\dots,Gr(f_m)).$$

Our previous considerations then imply that

$$\beta(A) = I_\psi(\beta(Gr(f_1)),\dots,\beta(Gr(f_m))).$$

By (β.1) and Remark 3.1, we can conclude that $\beta(A) \in \mathcal{EI}_0(N)$.

For the proof of the second assertion we confine ourselves again to the case of fixed points. So assume that we have a Σ^+ formula $\chi[P, \vec{x}]$ of L_N such that $B = I_\chi(-)$. Then, there exists an $r\Sigma_1^+$ formula $\theta[P, R_1, R_2, \vec{x}]$ of L_N with the property

$$B = I_\theta(Gr(0), Gr(S_u)).$$

Now we define

$$A := I_\theta(D_0, D_1)$$

and obtain

$$\beta(A) = I_\theta(\beta(D_0), \beta(D_1)) = I_\theta(Gr(0), Gr(S_u)) = B.$$

In addition, A is an element of $\mathcal{EI}_0(L)$ because of (β.2) and Remark 3.1.

II. $n \to n+1$. Using the same strategy, the assertions for $n+1$ follow immediately from the induction hypothesis. □

Using the same ideas, one can also prove that $\beta(\Sigma_1^+(\mathcal{K}, L)) = \Sigma_1^+(\beta(\mathcal{K}), N)$ if the class $\mathcal{K}$ contains $\mathcal{EI}_0(L)$.

Corollary 3.2. $\mathcal{EI}_{n+1}(L) = \Sigma_1^+(\mathcal{EI}_n(L) \cup \overline{\mathcal{EI}_n(L)}, L)$.

PROOF. If U_L is finite, then the assertion is trivial; otherwise, it follows from Theorem 3.1 and Theorem 3.2. □

4 Stratified programs

Now the ground is prepared for an easy characterization of the defining power of stratified programs with and without the stratified closed-world assumption *SCWA*. Related results have been obtained by Apt and Blair [4] who study the logical complexity of the supported models M_T of stratified programs T.

Stratified programs can be considered as special theories for iterated inductive definitions where the definition clauses are of very restricted form. We briefly review some basic notions and refer for more details to Apt, Blair, and Walker [2] and Lloyd [10]. A *level mapping* for L is a function α from the set Rel_L of the relation symbols of L to the natural numbers $\mathbb{N}$,

$$\alpha\colon Rel_L \to \mathbb{N}.$$

If R is an n-ary relation symbol of L and $\vec{t} = t_1, \ldots, t_n$ a sequence of L terms, then $\alpha(R(\vec{t}))$ and $\alpha(\neg R(\vec{t}))$ are defined to be the number $\alpha(R)$.

Definition 4.1. Let T be a normal program in L and α a level mapping for L.

(1) α is called *stratified* with respect to T if we have for all elements

$$F_1 \wedge \ldots \wedge F_n \ \rightarrow \ G$$

of T and all $1 \leq i \leq n$:

- $\alpha(F_i) \leq \alpha(G)$,
- $\alpha(F_i) < \alpha(G)$ provided that F_i is a negative literal.

(2) T is called *stratified* if there exists a level mapping for L which is stratified with respect to T.

(3) $\mathcal{S}_L^T$ denotes the set of all level mappings for L which are stratified with respect to T.

Example 4.1. The program consisting of the following four clauses

$$P(0), \quad Q(0), \quad Q(x) \rightarrow Q(f(x)), \quad Q(x) \wedge \neg P(x) \rightarrow R(g(x))$$

is stratified but not definite.

On the other hand, it is obvious that every definite program is stratified. The defining power of stratified programs therefore comprises that of the definite programs and is limited by the following well-known property of general recursively enumerable theories.

Remark 4.1. Let $\ulcorner\varphi\urcorner$ be the Gödel number of the L formula φ. If T is an arbitrary L theory, then there exists a subset A of the natural numbers which is recursively enumerable in $\{\ulcorner\varphi\urcorner : \varphi \in T\}$ such that we have for all $\psi \in B_L$:

$$T \vdash \psi \iff \ulcorner\psi\urcorner \in A.$$

In addition, for every relation symbol R of L, there exists a relation B_R which is recursively enumerable in $\{\ulcorner\varphi\urcorner : \varphi \in T\}$ such that we have for all $\vec{a} \in U_L$:

$$T \vdash R(\vec{a}) \iff \langle\beta(\vec{a})\rangle \in B_R.$$

In view of Theorem 3.1 and Theorem 3.2, this remark implies that all T-definable relations on U_L belong to $\mathcal{EI}_0(L)$. Not surprisingly, we therefore obtain a first theorem which characterizes the defining power of stratified programs.

Theorem 4.1. (Defining power of stratified programs).

(1) If T is a stratified program in L, then $Def_L(T)$ is a subset of $\mathcal{EI}_0(L)$.

(2) For every $A \in \mathcal{EI}_0(L)$, there exists a stratified — even definite — program T in L so that $A \in Def_L(T)$.

PROOF. The first assertion follows from the observation above. For the second, let A be an element of Σ^+-$IND(\emptyset, L)$. For notational simplicity we assume that there exists a binary relation $B \in \Sigma^+$-$FP(\emptyset, L)$ and a term $b_0 \in U_L$ so that

$$a \in A \iff \langle a, b_0 \rangle \in B$$

for all $a \in U_L$. The extension of our argument to the general case is straightforward. The relation B is the least fixed point of a Σ^+ formula $\varphi[Q, x, y]$ of L, i. e., $B = I_\varphi(-)$. By Lemma 2.5, this formula has a normal form

$$\begin{aligned}(\exists \vec{z})(x = s_1[\vec{z}] \wedge y = t_1[\vec{z}] \wedge \psi_1[Q, \vec{z}]) \vee \ldots \\ \vee (\exists \vec{z})(x = s_n[\vec{z}] \wedge y = t_n[\vec{z}] \wedge \psi_n[Q, \vec{z}]),\end{aligned}$$

where every $\psi_i[Q, \vec{z}]$ is a conjunction of atoms of the form $Q(\ldots)$. Now, define T to be the definite program

$$\{\psi_i[Q, \vec{z}] \to Q(s_i[\vec{z}], t_i[\vec{z}]) : i = 1, \ldots, n\} \cup \{Q(x, b) \to R(x)\}.$$

By some basic results on definite programs, it follows that

(1) $T \vdash Q(a, b) \iff \langle a, b \rangle \in B$,

(2) $T \vdash R(a) \iff a \in A$,

for all $a, b \in U_L$. □

A normal program T in L is stratified if and only if $\mathcal{S}_L^T \neq \emptyset$. In this case, we define for all relation symbols R of L:

$$\begin{aligned}\sigma_L^T(R) &:= \min\{\alpha(R) : \alpha \in \mathcal{S}_L^T\}, \\ \sigma_L(T) &:= \max\{\sigma_L^T(R) : R \text{ occurs in } L\}.\end{aligned}$$

σ_L^T is a stratified level mapping for T, called the *minimal stratification* of T. The number $\sigma_L(T)$ is denoted as the *stratified height* of T. Hence, a stratified program T is definite if and only if $\sigma_L(T) = 0$.

According to Reiter [14], the closed world $CWA(T, L)$ of a normal program T in L is usually defined as

$$CWA(T, L) := T \cup \{\neg F : F \in B_L \text{ and } T \nvdash F\}.$$

The closed-world assumption *CWA* is often considered as a problematic concept, especially since it often transforms (necessarily consistent) normal programs T into inconsistent theories $CWA(T, L)$. In the context of stratified programs, the situation can be significantly improved by replacing the general closed-world assumption *CWA* by the *stratified closed-world assumption SCWA*. An equivalent notion is introduced in [7] and denoted as *iterated closed-world assumption*. We prefer the name stratified closed-world assumption in order to distinguish it from the hierarchical closed-world assumption (to be introduced later) which is also generated by iterating the *CWA* in a suitable way.

Definition 4.2. Let T be a stratified program in L of height $\sigma_L(T) = m$. Then, we define for all $n \leq m + 1$:

$$\begin{aligned} SCWA_0(T, L) &:= T \\ SCWA_{n+1}(T, L) &:= \left\{ \begin{array}{l} SCWA_n(T, L) \cup \\ \{\neg F : F \in B_L,\ \sigma_L^T(F) = n \text{ and } SCWA_n(T, L) \nvdash F\}, \end{array} \right. \\ SCWA(T, L) &:= SCWA_{m+1}(T, L). \end{aligned}$$

Example 4.2. Assume that we have the stratified program

$$T = \{Q(0), \quad Q(x) \wedge \neg P(0) \rightarrow Q(S_u(x))\}$$

formulated in the language L_N. Then, the closed world of T,

$$CWA(T, L) = T \cup \{\neg F : F \in B_L \text{and } F \text{ different from } Q(0)\},$$

is inconsistent whereas the stratified closed world of T,

$$SCWA(T, L) = T \cup \{\neg P(a) : a \in U_N\},$$

is consistent. If we replace T by the logically equivalent stratified program

$$T' = \{Q(0), \quad Q(x) \wedge \neg Q(S_u(x)) \rightarrow P(0)\},$$

then the closed worlds of T and T' are the same whereas the stratified closed world of T' is the consistent theory

$$SCWA(T', L) = T \cup \{\neg Q(a) : a \in U_N \setminus \{0\}\} \cup \{\neg P(a) : a \in U_N \setminus \{0\}\}.$$

Hence, the stratified closed-world assumption *SCWA* is a more careful extension of the *CWA* — at the price of being sensitive to logical transformations of the basic theory. It is tailored for stratified programs in the sense that the theory $SCWA(T, L)$ is consistent for every stratified program T. The proof of the following lemma is straightforward.

Lemma 4.1. Let T be a stratified program in L and define

$$\Delta_L(T) := \{F \in B_L : SCWA_{\sigma_L^T(F)}(T, L) \vdash F\}.$$

Then, $\Delta_L(T)$ induces a Herbrand model of $SCWA(T, L)$.

For every stratified program T of height m and all $n \leq m$, we define the restriction $T \restriction n$ of T as the set of all formulas of T which do not contain relation symbols R of level $\sigma_L^T(R) > n$. Then T and $T \restriction n$ have the same proof-theoretic power with respect to relations of levels up to n, no matter whether we work with or without the *SCWA*.

Theorem 4.2. (Locality of stratified programs). Let T be a stratified program in L of height m. Then, we have for all $n \leq m$, all relation symbols R of level $\sigma_L^T(R) \leq n$ and all $\vec{a} \in U_L$:

(1) $T \vdash R(\vec{a}) \iff T \restriction n \vdash R(\vec{a})$,

(2) $SCWA_n(T, L) \vdash R(\vec{a}) \iff SCWA_n(T \restriction n, L) \vdash R(\vec{a})$,

(3) $SCWA(T, L) \vdash R(\vec{a}) \iff SCWA_n(T, L) \vdash R(\vec{a})$.

Now assume that T_0 and T_1 are stratified programs in L which have no relation symbols in common.

(4) If Q is a relation symbol which occurs in T_0 then we have for all $\vec{a} \in U_L$

$$SCWA(T_0 \cup T_1, L) \vdash Q(\vec{a}) \iff SCWA(T_0, L) \vdash Q(\vec{a}).$$

Proof. The first and second assertion can be checked easily. The third follows from Lemma 4.1. The fourth is proved by induction on the level $\sigma_L^{T_0}(Q)$ of the relation symbol Q. □

The following theorem answers the question about the structure of the relations on U_L which can be defined by stratified programs plus the stratified closed-world assumption. It also makes clear that the provability relation induced by the *SCWA* can be of arbitrary arithmetical complexity.

Theorem 4.3. (Defining power of the SCWA).

(1) *If T is a stratified program in L of height m, then $Def_L(SCWA(T,L))$ is a subset of $\mathcal{EI}_m(L)$.*

(2) *For every $A \in \mathcal{EI}_m(L)$, there exists a stratified program T in L of height m so that $A \in Def_L(SCWA(T,L))$.*

Proof. (1) Let T be a stratified program of height m. Then, an easy induction on $n \leq m$ shows:

(i) If R is a relation symbol of level $\sigma_L^T(R) = n$, then there exists a Σ^0_{n+1} relation A on the natural numbers so that we have for all $\vec{a} \in U_L$:

$$SCWA_n(T,L) \vdash R(\vec{a}) \iff \langle \beta(\vec{a}) \rangle \in A.$$

(ii) $\{\ulcorner \varphi \urcorner : \varphi \in SCWA_n(T,L)\}$ is a Σ^0_{n+1} subset of the natural numbers.

Hence (i) and Theorem 4.2 imply that, for every relation symbol R of level $\sigma_L^T(R) \leq m$, there exists a Σ_{m+1} relation A on the natural numbers satisfying

$$SCWA(T,L) \vdash R(\vec{a}) \iff \langle \beta(\vec{a}) \rangle \in A$$

for all $\vec{a} \in U_L$. Together with Theorem 3.1 and Theorem 3.2, we can therefore conclude that $Def_L(SCWA(T,L))$ is a subset of $\mathcal{EI}_m(L)$.

(2) The second assertion is proved by induction on m. Hence, let A be an element of $\mathcal{EI}_m(L)$. To keep the notation as simple as possible, we restrict ourselves to the discussion of the following special case (the extension of our arguments to full generality is then obvious):

(i) A is a section of a binary $B \in \Sigma^+\text{-}FP(\mathcal{EI}_{m-1}(L) \cup \overline{\mathcal{EI}_{m-1}(L)}, L)$, i.e., there exists a $b_0 \in U_L$ so that for all $a \in U_L$

$$a \in A \iff \langle a, b_0 \rangle \in B.$$

(ii) B is the least fixed point generated by the Σ^+ formula $\varphi[P,Q,R,x,y]$ and the unary $C \in \mathcal{EI}_{m-1}(L)$ and $D \in \overline{\mathcal{EI}_{m-1}(L)}$, i.e.,

$$B = I_\varphi(C,D).$$

We apply the induction hypothesis to C and $E :=\sim D$ and conclude that there are stratified programs T_C and T_E of height $m-1$ and relation symbols R_C and R_E satisfying

$$SCWA(T_C,L) \vdash R_C(a) \iff a \in C \tag{1}$$

$$SCWA(T_E, L) \vdash R_E(a) \iff a \in E \tag{2}$$

for all $a \in U_L$. Without loss of generality, we can assume that T_C and T_E have no relation symbols in common. From (2), we obtain with the stratified closed-world assumption that

$$SCWA(T_E, L) \vdash \neg R_E(a) \iff a \in D \tag{3}$$

for all $a \in U_L$. By Lemma 2.5, $\varphi[P, Q, R, x, y]$ has a normal form

$$\bigvee_{i=1}^{n} (\exists \vec{z})(x = s_i[\vec{z}] \wedge y = t_i[\vec{z}] \wedge \psi_i[P, Q, R, \vec{z}]),$$

where each $\psi_i[P, Q, R, \vec{z}]$ is a conjunction of atomic formulas $P(\ldots)$, $Q(\ldots)$ and $R(\ldots)$. Now we choose new relation symbols R_D, R_B and R_A and define

$$\begin{aligned} T \;:=\; & T_C \cup T_E \cup \{\neg R_E(x) \to R_D(x)\} \\ & \cup\; \{\psi_i[R_B, R_C, R_D, \vec{z}] \to R_B(s_i[\vec{z}], t_i[\vec{z}]) : i = 1, \ldots, n\} \\ & \cup\; \{R_B(x, b_0) \to R_A(x)\}. \end{aligned}$$

T is a stratified program of height $\sigma_L(T) \leq m$. Exploiting the locality of stratified programs and some simple properties of inductive definitions, we obtain for all $a, b \in U_L$:

$$SCWA(T, L) \vdash R_D(a) \iff a \in D, \tag{4}$$

$$SCWA(T, L) \vdash R_B(a, b) \iff \langle a, b\rangle \in B. \tag{5}$$

The direction "$\Longrightarrow$" of (5) is based on the fact that $B = I_\varphi(C, D)$ is a fixed point of $\varphi[P, Q, R, x, y]$ and therefore,

$$U_L \models \psi_i[B, C, D, \vec{a}] \implies \langle s_i[\vec{a}], t_i[\vec{a}]\rangle \in B \tag{6}$$

for all $\vec{a} \in U_L$. In order to establish the converse direction of (5), we recall that $B = I_\varphi(C, D) = I_\varphi^{<\omega}(C, D)$ and prove by induction on k:

$$\langle a, b\rangle \in I_\varphi^k(C, D) \implies SCWA(T, L) \vdash R_B(a, b). \tag{7}$$

If $\langle a, b\rangle \in I_\varphi^k(C, D)$, then there exist $\vec{c} \in U_L$ and $1 \leq i \leq n$ so that $a = s_i[\vec{c}]$, $b = t_i[\vec{c}]$ and

$$U_L \models \psi_i[I_\varphi^{k-1}(C, D), C, D, \vec{s}]. \tag{8}$$

By (4), the induction hypothesis and Lemma 2.8, we obtain

$$SCWA(T, L) \vdash \psi_i[R_B, R_C, R_D, \vec{c}] \tag{9}$$

and hence, by definition of T,

$$SCWA(T, L) \vdash R_B(a, b). \tag{10}$$

This completes the proof of (7). This equivalence, the definition of T and the definition of A finally yield

$$SCWA(T, L) \vdash R_A(a) \iff a \in A, \tag{11}$$

for all $a \in U_L$.

Hence, we have shown that the relation A is an element of $Def_L(SCWA(T, L))$ for some stratified program of height m. $\square$

Remark 4.2. This theorem can also be obtained by combining results of Apt and Blair [4] and Gelfond, Przymusinska and Przymusinski [7]. However, our approach is conceptually different and develops the definability theory of stratified programs from the more general point of view of inductive definability. We think that this provides a more perspicuous approach to stratified programs and reveals the close connections between stratified programs and inductive definitions.

5 Hierarchical programs

An alternative and important class of structured programs is provided by the class of the so-called hierarchical programs. As the stratified programs, they are defined by an easy-to-check syntactic condition which, however, is more restrictive than the requirements imposed on stratified programs.

Definition 5.1. Let T be a normal program in L and α a level mapping for L.

(1) α is called *hierarchical* with respect to T if we have for all elements

$$F_1 \wedge \ldots \wedge F_n \rightarrow G$$

of T and all $1 \leq i \leq n$:

$$\alpha(F_i) < \alpha(G).$$

(2) T is called *hierarchical* if there exists a level mapping for L which is hierarchical with respect to T.

(3) $\mathcal{H}_L^T$ denotes the set of all level mappings for L which are hierarchical with respect to T.

It is obvious that every hierarchical program is stratified. The following program T, on the other hand, is definite (and therefore stratified) but not hierarchical:

$$T = \{R(0), \quad R(x) \to R(S_u(x))\}.$$

If T is a hierarchical program in L, then $\mathcal{H}_L^T \neq \emptyset$, and we define for all relation symbols R of L:

$$\begin{aligned} \eta_L^T(R) &:= \min\{\alpha(R) : \alpha \in \mathcal{H}_L^T\}, \\ \eta_L(T) &:= \max\{\eta_L^T(R) : R \text{ occurs in } L\}. \end{aligned}$$

The situation corresponds to that of stratified programs: if T is a hierarchical program in L, then η_L^T is a hierarchical level mapping for T, called the *minimal hierarchical level mapping* of T; the number $\eta_L(T)$ is denoted as the *hierarchical height* of T. One has to observe, however, that, although every hierarchical program is stratified, the stratified height $\sigma_L(T)$ of a hierarchical program T may be different from its hierarchical height $\eta_L(T)$.

Hierarchical programs which are also definite provide an interesting subclass of the hierarchical programs and will be studied separately. Then, we turn to hierarchical programs with negation and finally to the hierarchical closed-world assumption.

Theorem 5.1. (Defining power of definite hierarchical programs).

(1) If T is a definite hierarchical program in L, then every element of $Def_L(T)$ is a term-definable relation in L.

(2) For every relation A on U_L which is term-definable in L, there exists a definite hierarchical program T in L of height 0 so that $A \in Def_L(T)$.

Proof. (1) Let T be a definite hierarchical program. The idea is to prove by induction on $\eta_T^L(R)$ that, for every relation symbol R, the set

$$A_R := \{\langle \vec{a} \rangle \in U_L^m : T \vdash R(\vec{a})\}$$

is term-definable. In the induction step, one has to make use of some well-known results on definite logic programs and the observation that A_R is Σ_1^+ in some relations which are term-definable by induction hypothesis. From Lemma 2.6 and Lemma 2.7 it follows that A_R is term-definable.

(2) Let A be a subset of U_L^n which is term-definable in L and of the form

$$A = \bigcup_{i=1}^{m} \{\langle t_{i,1}[\vec{a}], \ldots, t_{i,n}[\vec{a}] \rangle : \vec{a} \in U_L\}.$$

Then, A is definable by the program

$$T := \{Q(t_{i,1}[\vec{x}], \ldots, t_{i,n}[\vec{x}]) : i = 1, \ldots, m\}$$

which consists of atomic L formulas only. T is a definite hierarchical program in L of height 0. $\square$

The defining power of hierarchical programs with negation is particularly interesting. At first sight, it seems that hierarchical programs do not allow recursive definitions, since the same relation symbol must not occur on the left- and right-hand side of an implication. However, the following lemma shows that already, hierarchical programs of height 1 possess the same defining power as definite programs. The reason for this surprising result is the use of classical logic. If we work with a special form of resolution, the defining power of hierarchical programs with negation may collapse dramatically.

Lemma 5.1. For every definite program T in L, there exists a hierarchical program T^* in L of height 1 so that we have for all $F \in B_L$:

$$T \vdash F \iff T^* \vdash F.$$

Proof. Let $R_1, \ldots, R_m$ be the enumeration of the relation symbols occurring in T. Then, we choose new relation symbols $Q_1, \ldots, Q_m$ of corresponding arities and sufficiently many new variables $y_1, \ldots, y_n$ to carry through the following construction. If

$$\varphi \to R_i(t_1, \ldots, t_k)$$

is the element ψ of T, then we write $\overline{\varphi}$ for the formula which results from φ by replacing each relation symbol R_j by Q_j for $j = 1, \ldots, m$ and define ψ^* to be the formula

$$\overline{\varphi} \wedge \neg Q_i(t_1, \ldots, t_k).$$

Finally we set

$$T^* := \{\psi^* \to R_j(\vec{y}) : \psi \in T,\ j = 1, \ldots, m\} \cup \{Q_i(\vec{y}) \to R_i(\vec{y}) : i = 1, \ldots, m\}.$$

T^* is a hierarchical program in L of height 1, and it is easily shown that T and T^* prove the same elements of B_L. $\square$

Example 5.1. Let L be the language with the constant a and the unary function symbol f. Then, the definite program

$$T = \{\, R_0(a), \qquad R_0(x) \to R_1(f(x)), \qquad R_1(x) \to R_0(f(x)) \,\}$$

proves the same ground atoms as the hierarchical program T^* of height 1 given by the clauses:

$$\begin{array}{rlrl} \neg Q_0(a) & \to R_0(y), & Q_0(x) \wedge \neg Q_1(f(x)) & \to R_1(y), \\ \neg Q_0(a) & \to R_1(y), & Q_1(x) \wedge \neg Q_0(f(x)) & \to R_0(y), \\ Q_0(x) \wedge \neg Q_1(f(x)) & \to R_0(y), & Q_1(x) \wedge \neg Q_0(f(x)) & \to R_1(y). \end{array}$$

It is a consequence of this lemma that the class of hierarchical programs has the defining power of the class of definite program. Since, according to Theorem 4.1, the defining power of stratified programs is limited to $\mathcal{EI}_0$, we obtain the following result.

Theorem 5.2. (Defining power of hierarchical programs).

(1) If T is a hierarchical program in L, then $Def_L(T)$ is a subset of $\mathcal{EI}_0(L)$.

(2) For every $A \in \mathcal{EI}_0(L)$, there exists a hierarchical program T in L of height 1 so that $A \in Def_L(T)$.

Remark 5.1. It seems that any procedural approach to this form of defining power must be based on general resolution, which is never done in any Prolog-like environment.

Now, we adjust the definition of stratified closed-world assumption to the case of hierarchical programs in order to obtain the corresponding notion of hierarchical closed-world assumption. If follows the same idea as above but with the level function σ_L^T relaced by η_L^T.

Definition 5.2. Let T be a hierarchical program in L of height $\eta(T) = m$. Then, we define for all $n \leq m + 1$:

$$\begin{array}{rcl} HCWA_0(T, L) & := & T \\ HCWA_{n+1}(T, L) & := & \left\{ \begin{array}{l} HCWA_n(T, L) \cup \\ \{\neg F : F \in B_L,\ \eta_L^T(F) = n \text{ and } HCWA_n(T, L) \not\vdash F\}, \end{array} \right. \\ HCWA(T, L) & := & HCWA_{m+1}(T, L). \end{array}$$

In many aspects the hierarchical closed-world assumption is similar to the stratified closed-world assumption:

(1) There are hierarchical theories T such that $CWA(T, L)$ is inconsistent;

(2) $HCWA(T, L)$ is always consistent;

(3) the *HCWA* reflects a more careful closing process than the *CWA* and is sensitive to logical transformations of the underlying theory;

(4) Theorem 4.2 also holds for hierarchical programs.

But there is also one big difference. Whereas the defining power of stratified programs is enormously increased by adding the stratified closed-world assumption, this is not the case for the hierarchical closed-world assumption. Moreover, the following theorem shows that the defining power of hierarchical programs collapses if we allow closure under the *HCWA*. In order to prove it we need the following fixed point characterization of the hierarchical closed-world assumption.

Lemma 5.2. Let T be a hierarchical program in L. Then, $HCWA(T, L) \vdash F$ if and only if there exists a clause $\varphi \to G$ in T and a ground substitution σ such that $F = G\sigma$ and $HCWA(T, L) \vdash \varphi\sigma$.

Proof. Define Δ to be the collection of all atomic formulas $G\sigma \in B_L$ so that the following conditions are satisfied:

(1) σ is a ground substitution;

(2) $HCWA(T, L) \vdash \varphi\sigma$ for some clause $\varphi \to G$ in T.

Our lemma is established if we can show that Δ is the collection of elements of B_L which are provable in $HCWA(T, L)$, i.e.,

$$F \in \Delta \iff HCWA(T, L) \vdash F$$

for all $F \in B_L$. The implication from left to right is obvious, the implication from right to left is proved by induction on $\eta_L^T(F)$. Toward this end, assume $\eta_L^T(F) = n$ and $HCWA(T, L) \vdash F$. By the locality principle for hierarchical programs, it follows that $HCWA_n(T \restriction n, L) \vdash F$. Exploiting the induction hypothesis, it is then an easy exercise to show that

$$\Delta_n := \{H \in \Delta : \eta_L^T(H) \leq n\}$$

induces a Herbrand model of $HCWA_n(T \restriction n, L)$. This implies $F \in \Delta_n \subset \Delta$. □

Theorem 5.3. (Defining power of the HCWA).

(1) *If T is a hierarchical program in L, then every element of $Def_L(HCWA(T, L))$ is equationally definable in L.*

(2) *For every relation A on U_L which is equationally definable in L, there exists a hierarchical program T in L of height 1 so that $A \in Def_L(HCWA(T, L))$.*

PROOF. (1) It is sufficient to show — by induction on $\eta_L^T(Q)$ — that, for every relation symbol Q of L, there exists an equational formula $\varphi[\vec{x}]$ so that

$$HCWA(T, L) \vdash Q(\vec{a}) \iff U_L \models \varphi[\vec{a}]$$

for all $\vec{a} \in U_L$. If Q does not occur in T, then this assertion is trivially satisfied. Hence, let

$$\{\psi_i[\vec{R}, \vec{x}] \to Q(t_{i,1}[\vec{x}], \ldots, t_{i,n}[\vec{x}]) : i = 1, \ldots, m\}$$

be the definition of Q in T. Then, the hierarchical height of the relation symbols $\vec{R} = R_1, \ldots, R_k$ is smaller then $\eta_L^T(Q)$, and the induction hypothesis gives us equational formulas $\chi_1[\vec{x}], \ldots, \chi_k[\vec{x}]$ which correspond to $R_1, \ldots, R_k$:

$$HCWA(T, L) \vdash R_i(\vec{a}) \iff U_L \models \chi_i[\vec{a}]. \tag{12}$$

Because of the presence of the hierarchical closed-world assumption, it is immediate that

$$HCWA(T, L) \vdash \neg R_i(\vec{a}) \iff U_L \models \neg\chi_i[\vec{a}]. \tag{13}$$

Since each $\psi_i[\vec{R}, \vec{x}]$ is a conjunction of literals of the form $R_j(\ldots)$ and $\neg R_j(\ldots)$, we conclude that for all $1 \leq i \leq m$ and $\vec{a} \in U_L$,

$$HCWA(T, L) \vdash \psi_i[\vec{R}, \vec{a}] \iff U_L \models \psi_i[\chi_1, \ldots, \chi_k, \vec{a}], \tag{14}$$

where $\psi_i[\chi_1, \ldots, \chi_k, \vec{x}]$ indicates the result of substituting $\chi_j[\vec{t}]$ for every occurrence of $R_j(\vec{t})$ in $\psi_i[\vec{R}, \vec{x}]$. Now, we define

$$\varphi[\vec{x}] := \bigvee_{i=1}^{m} (\exists \vec{y})(x_1 = t_{i,1}[\vec{y}] \wedge \ldots \wedge x_n = t_{i,n}[\vec{y}] \wedge \psi_i[\chi_1, \ldots, \chi_k, \vec{y}]). \tag{15}$$

By Lemma 5.2, we have for all $\vec{a} \in U_L$:

$$HCWA(T, L) \vdash Q(\vec{a}) \iff U_L \models \varphi[\vec{a}]. \tag{16}$$

(2) Let A be defined by the equational formula $\varphi[\vec{x}]$, where $\vec{x} = x_1, \ldots, x_n$. According to Lemma 2.3, there exist strictly simple equality formulas $\psi_1[\vec{x}], \ldots, \psi_m[\vec{x}]$ such that

$$U_L \models \varphi[\vec{a}] \leftrightarrow \psi_1[\vec{a}] \vee \ldots \vee \psi_m[\vec{a}]$$

for all $\vec{a} \in U_L$. Now, choose a binary relation symbol EQ (for equality) and an n-ary relation symbol R. To a strictly simple equality formula $\theta[\vec{x}]$ of the form

$$(\exists \vec{y})\Big(\bigwedge_{i \in I} x_i = r_i[\vec{x}, \vec{y}] \wedge \bigwedge_{j \in J} x_{\sigma(j)} \neq s_j[\vec{x}, \vec{y}] \wedge \bigwedge_{k \in K} y_{\tau(k)} \neq t_k[\vec{x}, \vec{y}] \Big),$$

we associate the normal clause $\theta^*[\vec{x}]$ defined as

$$\bigwedge_{i \in I} EQ(x_i, r_i[\vec{x}, \vec{y}]) \wedge \bigwedge_{j \in J} \neg EQ(x_{\sigma(j)}, s_j[\vec{x}, \vec{y}]) \wedge \bigwedge_{k \in K} \neg EQ(y_{\tau(k)}, t_k[\vec{x}, \vec{y}]) \rightarrow R(\vec{x}).$$

Now, let T be the program

$$T := \{EQ(x, x)\} \cup \{\psi_i^*[\vec{x}, \vec{y}] : i = 1, \ldots, m\}.$$

Obviously T is a hierarchical program of height 1. Using Lemma 5.2, it follows that

$$U_L \models \varphi[\vec{a}] \iff HCWA(T, L) \vdash R(\vec{a})$$

for all $\vec{a} \in U_L$. □

Remark 5.2.

(1) Kunen [9] states a similar result for infinite languages and the negations as failure rule instead of the *HCWA*.

(2) Shepherdson ([16], Theorem 4) employs a similar construction in order to obtain related results for the *completion* of theories.

From work of Mal'cev [12] and Maher [11], one obtains the following decidability result for the validity of equational sentences over the corresponding Herbrand structure. It must not be confused with the (undecidable) notion of logical provability of an equational sentence.

Remark 5.3. Let φ be an equational formula of L which contains no free variables. Then, it is decidable whether $U_L \models \varphi$ or not.

This remark is interesting in our context, since it shows that the collection of equationally definable subsets of U_L is comparatively small. In particular, every equationally definable relation on U_L is recursive. A similar observation is also made in Apt and Blair [4], but there, it is a consequence of a different approach.

To end this paper, we consider the class of weakly hierarchical programs which is located — according to its syntactic definition — between the classes of hierarchical and stratified programs. With respect to its defining power, however, it corresponds to the stratified programs.

Definition 5.3. Let T be a normal program in L and α a level mapping for T.

(1) α is called *weakly hierarchical* with respect to T if we have for all elements

$$F_1 \wedge \ldots \wedge F_n \ \rightarrow \ G$$

of T and all $1 \leq i \leq n$:

- $\alpha(F_i) \leq \alpha(G)$,
- $\alpha(F_i) < \alpha(G)$ provided that F_i is a negative literal or $\alpha(G) \neq 0$.

(2) T is called *weakly hierarchical* if there exists a level mapping for L which is weakly hierarchical with respect to T.

(3) $\mathcal{WH}_L^T$ denotes the set of all level mappings for L which are weakly hierarchical with respect to T.

It is obvious that every hierarchical program is weakly hierarchical and every weakly hierarchical program is stratified. There are also weakly hierarchical programs which are not hierarchical, and stratified programs which are not weakly hierarchical, so that we have to deal with proper inclusions. The characterization of weakly hierarchical programs follows from Theorem 4.1 and Theorem 5.2.

Theorem 5.4. (Defining power of weakly hierarchical programs).

(1) If T is a weakly hierarchical program in L, then $Def_L(T)$ is a subset of $\mathcal{EI}_0(L)$.

(2) For every $A \in \mathcal{EI}_0(L)$, there exists a weakly hierarchical — even definite — program T in L so that $A \in Def_L(T)$.

The minimal hierarchical level mapping $w\eta_L^T$ of weakly hierarchical programs T and their weakly hierarchical heights $w\eta_L(T)$ are defined according to σ_L^T, $\sigma_L(T)$, η_L^T and $\eta_L(T)$. Based on $w\eta_L^T$, the weakly hierarchical closed-world assumption *W-HCWA*(T, L) of a weakly hierarchical program T is introduced analogously to Definition 4.2 and Definition 5.2. The defining power of weakly hierarchical programs with the weakly hierarchical closed-world assumption can be determined by making use of Corollary 3.2 and Theorem 4.3.

Theorem 5.5. (Defining power of the W-HCWA).

(1) If T is a weakly hierarchical program of height m, then $Def_L($ W-HCWA$(T, L))$ is a subset of $\mathcal{EI}_m(L)$.

(2) For every $A \in \mathcal{EI}_m(L)$, there exists a weakly hierarchical program T in L of height m so that $A \in Def_L($ W-HCWA$(T, L))$.

Open Questions.

(1) The first question refers to the choice of logic. As remarked above, many of our results — and especially Lemma 5.1 — are correct only since we worked with classical logic. Therefore, it could be interesting to find characterizations of the defining power of stratified and hierarchical programs in the presence of non-classical logics.

(2) Also the second question refers to Lemma 5.1. Our translation of definite programs into hierarchical programs does in general not provide allowed programs. Therefore, what is the defining power of allowed hierarchical programs?

References

1. Andreka, H., and Nemeti, I., The Generalized Completeness of Horn Predicate Logic as a Programming Language, *Acta Cybernetica*, 4(1):3–10 (1978).

2. Apt, K. R., Blair, H. A., and Walker, A., Towards a Theory of Declarative Knowledge, in: J. Minker (ed.), *Foundations of Deductive Databases and Logic Programming*, pages 89–148, Morgan Kaufmann, Los Altos, 1987.

3. Apt, K. R., Logic Programming, in: J. van Leeuwen (ed.), *Handbook of Theoretical Computer Science, Volume B*, chapter 10, pages 495–574, Elsevier, 1990.

4. Apt, K. R., and Blair, H. A., Arithmetic Classification of Perfect Models of Stratified Programs, *Fundamenta Informaticae*, 13:1–17 (1990). Addendum in vol. 14:339–343 (1991).

5. Barwise, J., *Admissible Sets and Structures: An Approach to Definability Theory*, Springer, Berlin, 1975.

6. Clark, K. L., Negation as Failure, in: H. Gallaire and J. Minker (eds.), *Logic and Data Bases*, pages 293–322, Plenum Press, New York, 1978.

7. Gelfond, M., Przymusinska, H., and Przymusinski, T., On the Relationship between Circumscription and Negation as Failure, *Artificial Intelligence*, 38:75–94 (1989).

8. Hinman, P. G., *Recursion-Theoretic Hierarchies*, Springer, Berlin, 1978.

9. Kunen, K., Answer Sets and Negation-as-Failure, in: J.-L. Lassez (ed.), *Logic Programming: Proceedings of the Fourth International Conference*, Melbourne University, pages 219–228, MIT Press, 1987.

10. Lloyd, J. W., *Foundations of Logic Programming*, Springer, Berlin, second, extended edition, 1987.

11. Maher, M. J., Complete Axiomatizations of the Algebras of Finite, Rational and Infinite Trees, in: *Third Annual Symposium on Logic in Computer Science (LICS)*, Edinburgh, Scotland, pages 348–357, 1988.

12. Mal'cev, A. I., Axiomatizable Classes of Locally Free Algebras of Various Types, in: *The Metamathematics of Algebraic Systems, Collected Papers*, chapter 23, pages 262–281, North-Holland, Amsterdam, 1971.

13. Moschovakis, Y. N., *Elementary Induction on Abstract Structures*, North-Holland, Amsterdam, 1974.

14. Reiter, R., On Closed World Data Bases, in: H. Gallaire and J. Minker (eds.), *Logic and Data Bases*, pages 55–76, Plenum Press, New York, 1978.

15. Shepherdson, J. C., Negation as Failure II, *Journal of Logic Programming*, 2(3):185–202 (1985).

16. Shepherdson, J. C., Language and Equality Theory in Logic Programming, Technical Report PM-88-08, University of Bristol, 1988.

17. Van Gelder, A., Negation as Failure Using Tight Derivation for General Logic Programs, in: J. Minker (ed.), *Foundations of Deductive Databases and Logic Programming*, pages 149–176, Morgan Kaufmann, Los Altos, 1987.