Towards a unified view of the hierarchy of logic program classes

Matthias Wendt

May 8, 2002

Submitted as general purpose project thesis

Under guidance of Dr. Pascal Hitzler

Supervised by Prof. Dr. Steffen Hölldobler

Knowledge Representation and Reasoning Group
Artificial Intelligence Institute
Department of Computer Science
Dresden University of Technology
Dresden, Germany

Abstract

A common approach to the handling of negation in logic programming semantics is to define a class of programs satisfying certain restrictions which then assure the applicability of certain semantic operators. Examples for this approach are the Φ-accessible programs [HS99] or the stratified programs [ABW88]. Research on logic programming semantics has yielded a complex hierarchy of logic program classes, due to more or less general restrictions. However, this hierarchy lacks a unified understanding. In the thesis we provide alternative characterizations of programs with total well-founded model and of weakly stratified programs. These new characterizations will employ level mappings, as they are used in the definition of locally hierarchical programs [Prz88]. Since level mappings are used in many characterizations of logic program classes, we hope that this will be another step towards a unified understanding of the hierarchy of logic program classes.

Contents

1 Introduction

In the history of logic programming and its theory, many efforts have been made to assign general logic programs, i.e. logic programs with negation, a both intuitive and unambiguous declarative semantics. Although very different approaches have been followed to achieve this goal, it has not yet been reached. One of these approaches, the "canonical model" approach, assigns a program a certain model and claims this to be the "intended" meaning of the program. This is usually done by defining a semantic operator associated to each logic program, whose fixed points are thought of as representing the intended meaning resp. its canonical models. However, like in the stable model approach [GL88], there may exist several fixed points of the semantic operator, none of which is preferable to the other ones. This unsatisfying ambiguity in the declarative semantics is then resolved by defining a restricted class of programs meeting some special conditions that assure the existence of least fixed points or even the uniqueness of fixed points of the semantic operator, thus making the declarative semantics unambiguous for the corresponding program class. Thus, as a counterpart to the canonical model approach, a hierarchy of logic program classes, corresponding to the proposed semantic operators, has emerged. For each of the semantics proposed for general logic programs, there exists a program class in the hierarchy, for which this semantics is uniquely determined. Each of the classes allows a different syntactic structure, e.g. recursion through positive atoms, to be used in the contained logic programs. Thus one can choose, dependent on what applications one wants to build, which semantics is the appropriate one.

The case is however not that easy: Due to the different ways in which the semantic operators resp. the program classes are defined – some of the semantic operators use program transformations that are not intuitively understandable, others do not –, these classes lack a unified understanding and therefore a unified comparison. One example of a class in this hierarchy not fully understood, is the class of weakly stratified programs [PP90]. Some mistakes concerning this class have appeared in the literature (e.g. in [BF91]), and we have reasons to believe, that this is caused by the above mentioned lack of a unified description. Some efforts in this direction have been made, and large parts of the hierarchy can now be understood by means of level mappings.

Level mappings are mappings that assign ordinals to the atoms in the Herbrand base associated with each program. Thus level mappings provide a convenient way of restricting the recursion used in the program clauses in that they can be used to compare the atoms in the head resp. the body of each clause. Usually one wants the level of the body literals to be (strictly) less than the level of the head of the clause, thus avoiding recursion. The restrictions imposed on the atoms can then be easily compared and provide an easy way of understanding to what extent recursion can be used in the resulting program class, see e.g. the definitions of acceptable programs [AP93] and locally hierarchical programs [Prz88]. Moreover, level mappings have been used in termination analysis and for topological investigations of the semantic operators underlying the program class in question.

As mentioned above, large parts of the hierarchy of logic programs can now be understood by means of level mappings. The problem we are facing now is, that the largest classes of programs that can be understood that way yet, are the Φ-accessible programs [HS99] and the locally stratifiable programs [Prz88]. Unfortunately, two generalizations of these two classes, the weakly stratified programs [PP90] and the programs having a total well-founded model [GRS91] resp. their finite pendants, the effectively stratifiable programs [BF91], that nevertheless can all be embedded in the hierarchy of logic programs, can not yet be expressed by definitions employing the notion of level mappings. This is somewhat unsatisfying, since the well-founded semantics is now one of the most accepted semantics for general logic programs.

The purpose of this thesis is thus to provide alternative characterizations for the aforementioned program classes with the help of level mappings and hence to gain a more intuitive approach to the "essence" of both the well-founded and the weakly stratified semantics for general logic programs. Most of the thesis will therefore be concerned with the proofs that our new characterizations are equivalent to the original ones in [GRS91] resp. [PP90].

We hope that our thesis will be another step towards a unified understanding of the hierarchy of logic program classes and that the alternative characterizations will prove to be useful both for intuitive understanding of the program classes in question resp. an intuitive comparison with other classes in the hierarchy as well as for ongoing research, e.g. topological investigations or implementation aspects.

The thesis will be structured as follows. In Section 2, we will provide the preliminaries and notation that is needed for understanding the topics presented in the thesis. Section 3 will be concerned with the alternative characterization of programs with total well-founded model resp. the proof that our definition really captures the essentials of the original definition of well-founded models. In Section 4, the same will be done for the weakly stratified programs. We conclude in Section 5 with a discussion of our results and their usefulness in different research directions.

Acknowledgement. I am very indebted to Dr. Pascal Hitzler, who introduced me to the complex of problems in logic programming and nonmonotonic reasoning, initiated the topic of this thesis, and patiently provided me with helpful discussions and corrections, which helped to increase especially the presentations of the proofs tremendously.

2 Preliminaries and Notation

In the following section we will recall the basic notions of logic programming that are used in the sequel. We basically stick to the notation of [Llo88]. A *normal logic program* is a finite set of clauses $\forall(A \leftarrow A_1 \wedge \cdots \wedge A_n \wedge \neg B_1 \wedge \cdots \wedge \neg B_m)$, which we will abbreviate by $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$. A, A_i and B_j, for $i = 1, \ldots, n$ and $j = 1, \ldots, m$, are atoms over a given first order language. We will call A the *head* and $A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ the *body* of the clause. We will sometimes also use operators with the same names, defined by the following equations:

$$\mathrm{head}(A \leftarrow L_1, \ldots, L_n) = A$$
$$\mathrm{body}(A \leftarrow L_1, \ldots, L_n) = L_1, \ldots, L_n.$$

The body of a clause may also be empty, these clauses are called *unit clauses* or *facts*. In abuse of notation we will sometimes regard $\mathrm{body}(A \leftarrow L_1, \ldots, L_n)$ as the set of body literals of the clause. We also use the refined notation of $\mathrm{body}^+(c)$ to denote the set of all positive literals in the body of the clause c, resp. $\mathrm{body}^-(c)$ for the set of negative literals in the body of c. A logic program is called *definite* or *positive*, if no negation occurs in the program.

For a given program P, we denote the set of ground instances of atoms occuring in the clauses of P by $\mathcal{B}_P$, the *Herbrand base*, and the ground instantiation of the program w.r.t. $\mathcal{B}_P$ by $\mathrm{ground}(P)$. For a given set L of literals, we denote the set that is obtained from L by taking the complement of each literal in L by $\neg \cdot L$.

Interpretations I in classical two-valued logic –two-valued interpretations for short – for a program P will be denoted by subsets of the Herbrand base, using the identification $\{A \in \mathcal{B}_P \mid I \models A\} = \{A \in \mathcal{B}_P \mid A \in I\}$. The set of all two-valued interpretations is denoted by $I_P = 2^{\mathcal{B}_P}$. For three-valued interpretations, i.e. interpretations in some logic with three truth values. we will use signed subsets of $\mathcal{B}_P$, i.e. subsets of $\mathcal{B}_P \cup \neg \cdot \mathcal{B}_P$. We then think of the atoms $A \in I \subseteq \mathcal{B}_P \cup \neg \cdot \mathcal{B}_P$ (resp. $\neg A \in I \subseteq \mathcal{B}_P \cup \neg \cdot \mathcal{B}_P$) as being true (resp. false) in I. If we have neither $A \in I$ nor $\neg A \in I$, we think of the atom as being undefined in I. A three-valued interpretation I is called *consistent* if for all atoms A not both $A \in I$ and $\neg A \in I$. In the following we will always assume interpretations to be consistent. The set of all consistent three-valued interpretations is denoted by $I'_P \subset 2^{\mathcal{B}_P \cup \neg \cdot \mathcal{B}_P}$. An interpretation I is called *total* if for each atom $A \in \mathcal{B}_P$ either $A \in I$ or $\neg A \in I$. Note that total three-valued interpretations can be identified with two-valued ones. We will identify $I^+ = \mathcal{B}_P \cap I$ resp. $I^- = \neg \cdot (I \setminus I^+)$ with the set of atoms that are true resp. false in the (three-valued) interpretation I.

A three-valued interpretation for a program P is called a *model for P* if all clauses of P are true under I w.r.t. Kleene's strong three-valued logic.

Kleene's strong three-valued logic is an extension of the classical logic in that it has an additional truth value $\mathbf{u}$, meaning 'undefined'. The truth tables for the connectives $\neg$ and $\wedge$ can be found e.g. in [Fit85].

The spaces we will be concerned with in the sequel all share the property of being *chain-complete partial orders*, usually abbreviated by cpo. A partially ordered set $(X, \leq)$ is called cpo, if i) there exists a *bottom element* $\perp$, such that $\perp \leq x$ for all $x \in X$, and ii) for all chains $(x_i)_{i \in I}$ we have $\sup_{i \in I}(x_i) \in X$, the latter giving rise to the term 'chain-complete'.

There exist two usual orderings on the set of all three-valued interpretations: the information ordering $\subseteq$, defined by $I_1 \subseteq I_2$ if and only if $I_1^+ \subseteq I_2^+$ and $I_1^- \subseteq I_2^-$, i.e. subset ordering. The other ordering is the truth ordering $\sqsubseteq$, given by $I_1 \sqsubseteq I_2$ if and only if $I_1^+ \subseteq I_2^+$ and $I_2^- \subseteq I_1^-$. The bottom element of the information ordering is $\emptyset$, whereas the bottom element of the truth ordering is $\neg \cdot \mathcal{B}_P$.

Now the spaces, we will be dealing with, are the following: The first of them is the set of all two-valued interpretations, endowed with the usual ordering, which is the standard subset inclusion, which we denote by $(I_P, \subseteq)$. For the case of three-valued interpretations, we then have two spaces, $(I'_P, \subseteq)$ and $(I'_P, \sqsubseteq)$, according to the two ordering relations defined above. Note that the information ordering in the three-valued case can be thought of as being a straightforward extension of the subset inclusion to signed subsets, so there is no problem with using the symbol $\subseteq$. All of these spaces are cpos. Note further, that $(I_P, \subseteq)$ is the restriction of $(I'_P, \subseteq)$ to the maximal elements (total three-valued interpretations), endowed with the $\sqsubseteq$ ordering.

An operator $T : X \to Y$, where $(X, \leq_X)$ and $(Y, \leq_Y)$ are cpos, is called *monotonic*, if it is order-preserving, i.e. if for $x, y \in X$ we have $T(x) \leq_Y T(y)$, whenever $x \leq_X y$. Further, T is called *continuous*, if it is suprema-preserving, i.e. if for each chain $(x_i)_{i \in I}$ we have, that $(T(x_i))_{i \in I}$ is again a chain and $T(\sup_{i \in I}(x_i)) = \sup_{i \in I}(T(x_i))$. One defines the powers of a operator that is monotonic w.r.t. an ordering $\leq$ as follows: $T \uparrow_{\leq} 0 = \perp_{\leq}$, $T \uparrow_{\leq} (\alpha + 1) = T(T \uparrow_{\leq} \alpha)$ for successor ordinal α, and $T \uparrow_{\leq} \alpha = \sup_{\leq}\{T \uparrow_{\leq} \beta | \beta < \alpha\}$ for limit ordinal α. We will omit the subscripts, if it is clear from the context, which ordering relation is used. Ordinals, as in the above transfinite definition, will be denoted by greek letters. The first infinite ordinal is denoted by ω. A level mapping for a program is a mapping $|\cdot| : \mathcal{B}_P \to \alpha$, where α is an ordinal. This level mapping can easily be extended to all ground literals by taking $|\neg A| = |A|$ for all $A \in \mathcal{B}_P$.

There are two famous fixed point theorems for operators on cpos in the literature: The one of Knaster-Tarski assures the existence of a least fixed point for a monotonic operator $T : X \to X$ on a cpo X. Further, this least fixed point can be obtained as a (transfinite) power of T. If T is in addition continuous, then $T \uparrow \omega$ is the least fixed point of T, by the fixed point theorem of Kleene.

For a given logic program P we will denote the immediate consequence operator by $T_P : I_P \to I_P$ [ABW88]. For a program P and a two-valued interpretation I, $T_P(I)$ is the set of all atoms $A \in \mathcal{B}_P$, such that there exists a clause $A \leftarrow L_1, \dots, L_n \in \text{ground}(P)$ and $I \models L_1, \dots, L_n$. Note that T_P is monotonic and continuous for definite programs. The fixed points of T_P are called supported models of P [ABW88]. A supported model is a model of the Clark completion of a normal logic program P [Cla78]. The Clark completion is denoted by $\text{comp}(P)$. A model I is supported if for every $A \in I$ there exists a clause $A \leftarrow \text{body} \in P$ such that $I \models \text{body}$.

We now repeat the definition of the dependency graph of a logic program.

2.1 Definition (Dependency Graph [ABW88]) The vertices of the dependency graph G_P of a program P are all ground atoms appearing in $\text{ground}(P)$. The edges of G_P are defined as follows. For every clause $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$, in P and for every $1 \leq i \leq n$ a positive directed edge from A_i to A is included in G_P and for every $1 \leq j \leq m$ a negative directed edge from B_j to A is included in G_P. The *dependency relations* $<$ and $\leq$ between ground atoms of P are defined by:

1. $A \leq B$ if and only if there is a directed path from A to B.
2. $A < B$ if and only if there is a directed path from A to B passing through at least one negative edge.

The following two notions were introduced in [AP93]: For two atoms A and B occuring in P, we say A *refers to* B, if there is a directed edge in the atom dependency graph of P. We further say, that A depends on B, if (A, B) is in the transitive closure of the *refers to*-relation, i.e. $A \geq B$.

We now at last define two properties of logic programs, that will be often used in the sequel. The first of them is the notion of a stratified program due to [ABW88].

2.2 Definition Let P be a logic program. P is said to be *stratified* if there is a partition $P = P_1 \cup \dots \cup P_m$ of P such that the following two conditions hold for $i = 1, \dots, m$:

1. If a predicate symbol A occurs positively in the body of a clause in P_i, then all clauses with head A are contained in $\bigcup_{j \leq i} P_j$.
2. If a predicate symbol A occurs negatively in the body of a clause in P_i, then all clauses with head A are contained in $\bigcup_{j < i} P_j$.

Now follows the definition of stable models, introduced in [GL88]:

2.3 Definition Let P be a logic program. For an interpretation I, we obtain the program P_I, by applying the following two rules:

1. Delete each clause from ground(P) that has a negative literal $\neg B$ in the body with $B \in I$.

2. Delete all negative body literals from the bodies of the remaining rules.

Now the operator GL_P, the *Gelfond-Lifschitz operator*, is defined as $\mathrm{GL}_P\,(I) = T_{P_I} \uparrow \omega$. If now I is the least Herbrand model of P_I – or, equivalently, if I is a fixed point of GL_P –, it is called a *stable model* of P.

Throughout the thesis we will use the term *semantics* for *denotational semantics*, *program* for normal logic program and sometimes refer to clauses as rules.

3 A new characterization of programs with total well-founded model

In the following section, we will be mainly concerned with our new characterization of the programs having a total well-founded model. As we will see later on, this is in fact the easier part of the thesis, since the characterization of the weakly stratified programs is much more complicated both to state and to prove it to be equivalent to the original notion of [PP90].

We will characterize the programs with total well-founded model by employing level mappings. This will yield an intuitive understanding of this program class. Moreover, it is likely that this characterization may be easier to carry over to e.g. the setting of disjunctive logic programs resp. the well-founded semantics belonging to them. Another possibility is the application of metric methods, an approach closely connected to the use of level mappings, to the operators yielding the well-founded semantics, as done e.g. in [Hei01].

Additionally, we will further investigate the effectively stratifiable programs, that were introduced in [BF91]. These programs were proven to be equivalent to those having a total well-founded model. Unfortunately, this was done only for the case of finite propositional programs. A generalization to instantiations of first-order programs resp. infinite propositional programs was proposed in this paper. However, the definition given there lacks a formal argument for the existence resp. uniqueness of the greatest lower bound employed for the limit case, see also [BF91, Section 4]. It is left unclear resp. unproven which ordering relation is used for this greatest lower bound, or if this greatest lower bound does always exist. In dealing with the effectively stratifiable programs, we therefore stick to the finite propositional case, in order to avoid trouble with greatest lower bounds as in [BF91, Section 4].

The section will begin with a repetition of the main technical details and preliminaries for the well-founded models resp. the effectively stratifiable programs before stating our characterization and proving it to be equivalent to the original definition. The reader that is already familiar with all of these definitions and theorems may therefore skip the first two subsections.

3.1 Preliminaries for the well-founded models

We now give the main definitions for the well-founded models, that are taken from [GRS91] with some modifications to fit our notation. Since the well-founded model may not be total, we assume interpretations to be three-valued ones in the following.

The first notion we need, is that of satisfied resp. falsified clauses, as it was defined in [GRS91, Definition 2.5].

3.1 Definition Let P be a program. A clause $A \leftarrow \text{body} \in \text{ground}(P)$ is called *satisfied* in a (partial) interpretation I, if $I \models A$ or $\neg L \in I$, i.e. L is false in I, for some $L \in$ body. It is called *falsified*, if $\neg A \in I$ and $I \models L$ for all $L \in$ body. Note that the first two notions emerge straightforwardly from the truth table of the implication. If further $\neg A \in I$ and $I \not\models \neg L$ for all $L \in$ body, i.e. the body of the clause is not false in I, we say the clause is *weakly falsified* in I.

Now we repeat the notion of an unfounded set, that can also be found in [GRS91, Definition 3.1]:

3.2 Definition (Unfounded Sets) Let P be a program and I be an interpretation. We say $U \subseteq \mathcal{B}_P$ is an *unfounded set w.r.t.* I if each atom $A \in U$ satisfies the following condition: For each instantiated rule $r \in \text{ground}(P)$ with $\text{head}(r) = A$, (at least) one of the following holds:

1. Some (positive or negative) body literal L is false in I, i.e. $\neg L \in I$.

2. Some positive body literal occurs in U.

The unfounded sets have an interesting property, which will be used in the equivalence proofs later on and that is given in the following lemma [GRS91, Lemma 3.2]:

3.3 Lemma Let I be a partial interpretation consisting of positive literals Q and negative literals $\neg \cdot S$. If I does not weakly falsify any clause in $\text{ground}(P)$, then S is an unfounded set w.r.t. Q.

We now define the constituting operators in the definition of the (partial) well-founded model:

3.4 Definition Transformations $\mathbf{T}_P$, $\mathbf{U}_P$ and $\mathbf{W}_P$ on the set of partial interpretations for P are defined as follows:

- $A \in \mathbf{T}_P\,(I)$ if and only if there is some rule $r \in \text{ground}(P)$ such that $\text{head}(r) = A$ and for each body literal $L \in \text{body}(r)$ we have $I \models L$.

- $\mathbf{U}_P(I)$ is the greatest unfounded set of P w.r.t. I.

- $\mathbf{W}_P(I) = \mathbf{T}_P(I) \cup \neg \cdot \mathbf{U}_P(I)$.

The greatest unfounded set is the union of all sets that are unfounded w.r.t. I. Note that the transformations $\mathbf{T}_P$, $\mathbf{U}_P$ and $\mathbf{W}_P$ are monotonic w.r.t. set inclusion, i.e. on the cpo $(I'_P, \subseteq)$, as was stated in [GRS91, Lemma 3.3].

The transformation $\mathbf{W}_P$ is now used for the definition of (partial) well-founded models in [GRS91, Definition 3.4]:

3.5 Definition (Well-founded partial models) In the following, let α be a countable ordinal. The sets I_α and I^∞, whose elements are literals over the Herbrand base of a program P, are defined recursively by:

1. For a limit ordinal α,
$$I_\alpha = \bigcup_{\beta<\alpha} I_\beta.$$
Note that 0 is a limit ordinal and $I_0 = \emptyset$.

2. For a successor ordinal $\alpha = \gamma + 1$,
$$I_{\gamma+1} = \mathbf{W}_P(I_\gamma).$$

3. Finally, define
$$I^\infty = \bigcup_\alpha I_\alpha,$$
i.e. I^∞ is the least fixed point of the operator $\mathbf{W}_P$ on the cpo $(I'_P, \subseteq)$, which by monotonicity of $\mathbf{W}_P$ must exist according to the Knaster-Tarski fixed point theorem.

I^∞ is called the *(partial) well-founded model* of the program P. Moreover, if I^∞ is a total model, we call I^∞ the *total well-founded model* of the program P.

Throughout the sequel we will sometimes write $\mathbf{W}_P \uparrow \alpha$ for I_α, since it is straightforward from the above definition to think of I_α as the α-th power of $\mathbf{W}_P$.

Finally, we state the relationship between the well-founded model and the stable models of a program P (recall the definition of stable models from Section 2). This result was proven in [GRS91, Theorem 5.4]:

3.6 Theorem Let P be a program and I be a total model for P. Then I is a stable model of P if and only if it is a fixed point of $\mathbf{W}_P$.

From this, the following corollary can be obtained [GRS91, Corollary 5.7]:

3.7 Corollary The (partial) well-founded model of a program P is a subset of every stable model of P.

This follows easily from the fact, that the (partial) well-founded model of P is the least fixed point, i.e. all fixed points (including the stable models of P) must be greater w.r.t. $\subseteq$, hence P is a subset of every stable model of P.

3.2 Preliminaries for effectively stratifiable programs

We will now give the main definitions for the notion of effectively stratifiable programs. As noted earlier, the effectively stratifiable programs were only defined for the special case of finite propositional programs. Thus in the following, the term "program" refers to finite propositional programs, that can be thought of as instantiations of first-order programs without function symbols.

For introducing the notion of effectively stratifiable programs, we will use the "operational" definition via the EFF-operator, for the fixed point definition via the default consequence operator E_P we refer to [BF91, Section 3]. We first need the notions of defined, undefined and potentially defined propositions, as given in [BF91, Definition 4.1]. In the following let P^+ denote the greatest subprogram of P containing only definite clauses.

3.8 Definition Let P be a logic program. Then *defined* propositions in P are recursively defined as follows:

1. if $A \leftarrow$ is in P, then A is defined in P,
2. if there exists a clause $r \in P^+$ with $\text{head}(r) = A$ and for all $B \in \text{body}(r)$, B is defined in P, then A is defined in P, and
3. all defined propositions in P are generated by applying the two above rules.

The set of defined propositions in P is denoted by $\text{Def}(P)$.

The *potentially defined* propositions in P are recursively defined as follows:

1. if $A \leftarrow$ is in P, then A is potentially defined in P,
2. if there exists a clause $r \in P$ with $\text{head}(r) = A$ and for all $B \in \text{body}^+(r)$, B is potentially defined in P, then A is potentially defined in P, and
3. all potentially defined propositions in P are generated by applying the two above rules.

The set of potentially defined propositions in P is denoted by $\text{Potdef}(P)$. A proposition is *undefined* in P if and only if A is not potentially defined in P. The set of undefined propositions in P is denoted by $\text{Undef}(P)$.

Note that since $\text{Def}(P) \subseteq \text{Potdef}(P)$ we have $\text{Def}(P) \cap \text{Undef}(P) = \emptyset$. Note also that the defined propositions are thought to be true, undefined propositions to be false and potentially defined propositions, that are not defined, to be undefined in the corresponding stable model (see [BF91]). Now these truth values are used to reduce the program, as done in the following.

The operations Reduce and Simplify are defined in [BF91, Definition 4.7], the operator EFF in [BF91, Definition 4.9].

3.9 Definition (Effective operator EFF**)** Let P be a logic program and L be a literal. Then

$$\text{Reduce}(P, L) = \{r | s \in P, \text{body}(r) = \text{body}(s) \setminus \{L\} \text{ and } \text{head}(r) = \text{head}(s)\}$$

$$\text{Simplify}(P, L) = P \setminus \{r | L \in \text{body}(r)\}$$

The extension of these definitions to sets of literals in the second argument is straightforward and omitted here, the detailed definition can be found in [BF91, Definition 4.8]. Now, the effective operator, denoted by EFF, is defined by:

$$\begin{aligned} \text{EFF}(P) = \text{Simplify}(\text{Reduce}(P, \text{Def}(P) \cup \neg \cdot \text{Undef}(P)), \\ \neg \cdot \text{Def}(P) \cup \text{Undef}(P)) \end{aligned}$$

Before we can consider powers of this operator on logic programs, we have to define an ordering relation on logic programs, as done in [BF91, Definition 4.13]. We will see later on that the operator EFF is monotonic w.r.t. the ordering relation on logic programs defined below, so the definition of the powers of the EFF-operator w.r.t. this ordering relation is straightforward.

3.10 Definition (Subsumption relation) Let r and s be two clauses and P and Q be two logic programs. Then

1. $r \preccurlyeq s$ if and only if $\text{head}(r) = \text{head}(s)$ and $\text{body}(r) \subseteq \text{body}(s)$. We say r *subsumes* s.
2. $P \preccurlyeq Q$ if and only if for each clause $r \in P \setminus Q$ there exists a clause $s \in Q \setminus P$ such that $r \preccurlyeq s$.

Now we can define the powers of the EFF-operator by setting $\text{EFF}^0(P) = P$ and $\text{EFF}^{n+1}(P) = \text{EFF}(\text{EFF}^n(P))$ for any natural number $n \in \mathbb{N}$. It was shown in [BF91, Property 4.14], that this sequence of powers of the EFF-operator is decreasing, hence yields a fixed point. Note that we do not need to work over ordinals, i.e. we do not need to define a limit case here, since we only consider finite propositional programs.

The notion of effective stratifiability is now easily defined using the powers of EFF, [BF91, Definition 4.15].

3.11 Definition Let P be a finite propositional program. P is *effectively stratifiable* if and only if there exists n such that $\text{EFF}^n(P)$ is stratified.

The following theorem describes the relationship between the models of effectively stratifiable programs – it was proven in [BF91, Theorem 5.9, Theorem 5.15].

3.12 Theorem Let P be a finite propositional logic program. P is effectively stratifiable if and only if the (partial) well-founded model of P is total. Moreover, for effectively stratifiable programs, unique stable model and well-founded model coincide.

3.3 An alternative characterization of programs having a total well-founded model

In the following section an alternative characterization of programs having a total well-founded model will be given. The programs satisfying the conditions of this definition will be denoted "programs, that are *well-founded w.r.t. a model I and a level mapping $|\cdot|$*". This term must not be confused with that of a program with a total well-founded model: Programs that are well-founded w.r.t. a model I and a level mapping $|\cdot|$ do in fact capture the same program class in the hierarchy, but nevertheless this equality is yet to be shown. The only purpose of this new name is to distinguish between the definition of programs with total well-founded models as in [GRS91] and our alternative definition of the same program class, since their equality is not obvious and to be proven in this chapter. Thus this alternative terminology will only be used until the equality of both program classes is established.

The structure of the definition itself is inspired by that of the Φ-accessible programs from [HS99], and can clearly be seen to be a generalization of these programs. It can be referred to as being semi-syntactic, since the main plot in the definition is to impose restrictions on the structure of the clauses of a logic program, but this can only be done with the help of an (a-priori given) model. As such, this approach is akin to the definition of acceptable programs in [AP93].

The definition is now as follows:

3.13 Definition A program is *well-founded w.r.t. a model I and a level mapping* $|\cdot| : \mathcal{B}_P \to \gamma$ if and only if for every atom $A \in \mathcal{B}_P$ either one of the following conditions holds:

1. There exists a clause $A \leftarrow L_1, \ldots, L_n \in \text{ground}(P)$ such that $I \models L_1, \ldots, L_n$ and $|A| > |L_i|$ for all $i = 1, \ldots, n$.
2. $I \not\models A$ and for each clause $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m \in \text{ground}(P)$, (at least) one of the following conditions holds:
   (a) There exists $i = 1, \ldots, n$ such that $I \not\models A_i$ and $|A| \geq |A_i|$.
   (b) There exists $j = 1, \ldots, m$ such that $I \models B_j$ and $|A| > |B_j|$.

Note that this definition implies that I is a supported model: If for an atom $A \in \mathcal{B}_P$ we have $A \in I$, then only the first condition can be satisfied, thus there exists a clause $A \leftarrow$ body with $I \models$ body, which is the condition for supportedness of a model.

We now are going to show the equality of Definition 3.13 with the programs with total well-founded models from Definition 3.5. The first thing to do is to show that every program with total well-founded model is well-founded w.r.t. a model I and a level mapping $|\cdot|$. In fact we will show that I and $|\cdot|$ can be chosen, such that I is the total well-founded model of the program and that $|\cdot|$ meets some minimality property, giving it a kind of canonicity. This *canonical level mapping*, that will be denoted by $|\cdot|_P$, is obtained by the following construction:

3.14 Construction (Canonical level mapping) Every atom $A \in \mathcal{B}_P$ is assigned its level w.r.t. $\mathbf{W}_P$, i.e. $|A|_P = \alpha$ if and only if α is the least ordinal such that $A \in I_{\alpha+1}$ or $\neg A \in I_{\alpha+1}$.

Note that we have done some little "shift", since an atom $A \in \mathcal{B}_P$ is not assigned its "real" level w.r.t. $\mathbf{W}_P$, but its predecessor. The purpose of this is, that the so-obtained level mapping has a minimality property it would not have without the shift. However, this shift does no harm. The least ordinal α with $A \in I_\alpha$ or $\neg A \in I_\alpha$ can not be a limit ordinal, because from the construction of the well-founded model we have for a limit ordinal α the recursion equation $I_\alpha = \bigcup_{\beta<\alpha} I_\beta$, thus if $A \in I_\alpha$ (resp. $\neg A \in I_\alpha$) there must exist some ordinal $\beta < \alpha$ with $A \in I_\beta$ (resp. $\neg A \in I_\beta$), giving the contradiction that α is the least ordinal with $A \in I_\alpha$ or $\neg A \in I_\alpha$. Thus the "real" level of an atom is always a successor ordinal, and has therefore a predecessor. This predecessor is now the canonical level of the atom.

The next important property our level mapping must satisfy (and it does) is the totality, guaranteed by the following lemma:

3.15 Lemma Let P be a program with total well-founded model. The canonical level mapping from Construction 3.14 is total, i.e. for all atoms $A \in \mathcal{B}_P$ we have an ordinal α such that $|A|_P = \alpha$.

Proof: P has a total well-founded model, i.e. for each atom $A \in \mathcal{B}_P$ there must exist a least ordinal α such that $A \in I_{\alpha+1}$ or $\neg A \in I_{\alpha+1}$ (otherwise the model would not be total) and from Construction 3.14 we have $|A|_P = \alpha$. ■

Now having constructed the canonical level mapping and established its totality we are ready to prove the first direction, expressed by the following proposition.

3.16 Proposition If a program P has a total well-founded model, then it is also well-founded w.r.t. a model I and a level mapping $|\cdot|$, where the model I is the total well-founded model and the level mapping $|\cdot|$ is the canonical level mapping $|\cdot|_P$ from Construction 3.14.

Proof: For the following proof, let P be a program with total well-founded model I and $|\cdot|_P$ be the canonical level mapping. We are going to do the proof by transfinite induction on the ordinal index of the model I_α the atom $A \in \mathcal{B}_P$ first appears in. This index can – by the considerations made above – not be a limit ordinal and thus we do not have to consider the limit case, only the successor case. Now assume $\alpha + 1$ to be the successor of the level of the atom $A \in \mathcal{B}_P$ and the proposition holds for all atoms with level less than α. We distinguish two cases:

(1) If $I \models A$, we have that $\alpha + 1$ is the least ordinal such that $A \in I_{\alpha+1}$. But we can have $A \in I_{\alpha+1}$ only (by application of $\mathbf{T}_P$) if there is some clause $A \leftarrow L_1, \dots, L_n \in \text{ground}(P)$ such that all literals $L_1, \dots, L_n$ are true in I_α. From this we have $|L_i|_P < \alpha = |A|_P$ for all $i = 1, \dots, n$.

(2) If otherwise $I \not\models A$, we have that $\alpha + 1$ is the least ordinal such that $\neg A \in I_{\alpha+1}$. From Definition 3.5 we then have $A \in \mathbf{U}_P(I_\alpha)$. We now distinguish two cases according to the two conditions for A being in $\mathbf{U}_P(I_\alpha)$ (for each clause with head A at least one of them must hold, see Definition 3.2). Let $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$ be a clause in ground(P).

(i) Some (positive or negative) literal in the body must be false in I_α. This implies $\neg A_i \in I_\alpha$ for some $i = 1, \dots, n$ (resp. $B_j \in I_\alpha$ for some $j = 1, \dots, m$), yielding $|A_i|_P < \alpha = |A|_P$ (resp. $|B_j|_P < \alpha = |A|_P$).

(ii) Some positive literal A_i in the body occurs in $\mathbf{U}_P(I_\alpha)$. Assume now w.l.o.g. $\neg A_i \notin I_\alpha$. Since $A_i \in \mathbf{U}_P(I_\alpha)$ we have $\neg A_i \in I_{\alpha+1}$, yielding $|A_i|_P = \alpha = |A|_P$.

We thus have via transfinite induction the conditions of Definition 3.13 holding for all atoms $A \in \mathcal{B}_P$ and the proof is complete. ■

After having proved that every program having a total well-founded model is in fact well-founded w.r.t. its (total) well-founded model and the canonical level mapping $|\cdot|_P$, we are now going to show that the canonical level mapping is minimal.

3.17 Proposition Let P be a program with total well-founded model I. The level mapping $|\cdot|_P$ from Construction 3.14 is minimal, i.e. if $|\cdot|$ is a level mapping satisfying the conditions from Definition 3.13 we have for every atom $A \in \mathcal{B}_P$ that $|A|_P \leq |A|$.

Proof: Note that the "shift" in Construction 3.14 implies that all atoms $A \in \mathbf{T}_P(\emptyset)$ or $A \in \mathbf{U}_P(\emptyset)$ have $|A|_P = 0$. Now assume w.l.o.g. $A \in \mathcal{B}_P$ to be an atom such that A is minimal w.r.t. $|\cdot|_P$ and $|A|_P = \alpha$. Assume further that $|A| < |A|_P$. We distinguish two cases:

(1) If $I \models A$ we have α is the least ordinal such that $A \in I_{\alpha+1}$ and we have $A \in \mathbf{T}_P(I_\alpha)$. We now consider the two subcases of α being successor resp. limit ordinal.

(i) If α is a successor ordinal, we have from $A \in \mathbf{T}_P(I_\alpha)$ that there must exist a clause $A \leftarrow L_1, \dots, L_n \in \text{ground}(P)$ with $L_i \in I_\alpha$ for all $i = 1, \dots, n$ and $|L_i|_P \leq \alpha - 1$. Furthermore there can not exist any clause in ground(P)

with head A and all body literals having level strictly less than $\alpha - 1$. If such a clause existed, α would not be the least ordinal such that $A \in I_{\alpha+1}$, by definition of $\mathbf{T}_P$. We thus have, in every clause with head A whose body is true in I_α, a literal with $|L_i|_P \geq \alpha - 1$ and by our assumption $|L_i| \geq |L_i|_P$. If we now take $|A| < \alpha$ we do not have any clause satisfying the condition 1 of Definition 3.13. This contradicts the assumption that there exists a level mapping with $|A| < |A|_P$ satisfying the conditions of Definition 3.13.

(ii) If otherwise α is a limit ordinal, from Definition 3.5 we have that for every ordinal $\beta < \alpha$ we have, in every clause with head A whose body is true in I_α, some literal L_i with $|L_i|_P > \beta$ (this is the cause for α to be a limit ordinal). Now if we take $|A| < \alpha$ we have, in every clause with head A whose body is true under I_α, a literal with $|L_i| \geq |L_i|_P > |A|$, thus no clause with head A satisfies the condition 1 from Definition 3.13. This contradicts the assumption that there exists a level mapping with $|A| < |A|_P$ satisfying the required conditions.

(2) If $I \not\models A$, then we have that α is the least ordinal such that $\neg A \in I_{\alpha+1}$ and therefore $A \in \mathbf{U}_P(I_\alpha)$. We again distinguish the successor and the limit case:

(i) If α is a successor ordinal, we have from $A \in \mathbf{U}_P(I_\alpha)$ that there must exist a clause such that $|L_i|_P \geq \alpha - 1$ for all body literals L_i false in I, otherwise α would not be the least ordinal such that $\neg A \in I_{\alpha+1}$, by definition of $\mathbf{U}_P$. Note that the positive literals A_i in that clause with $\neg A_i \notin I_\alpha$ and $A_i \in \mathbf{U}_P(I_\alpha)$ do have level mapping $|A_i|_P = \alpha$. If we now take $|A| < \alpha$ we have by the assumption that $|L_i| \geq |L_i|_P$ for all body literals L_i that the condition 2 in Definition 3.13 does not hold for all clauses. This contradicts the assumption that there exists a suitable level mapping such that $|A| < |A|_P$.

(ii) If otherwise α is a limit ordinal, from the Definition 3.5 we have that for every ordinal $\beta < \alpha$ there exists a clause with head A with all false body literals L_i having level mapping $|L_i|_P > \beta$. Now, if we take $|A| < \alpha$, then by our assumption $|L_i| \geq |L_i|_P$ not all clauses can satisfy the condition 2 of Definition 3.13. This contradicts the assumption that there exists a suitable level mapping such that $|A| < |A|_P$.

This establishes the minimality of the canonical level mapping $|\cdot|_P$. ■

Having now finished this little interludium (this proof is not really necessary to establish the equality), we are going to prove the converse direction, i.e. we prove that every program that is well-founded w.r.t. a model I and a level mapping $|\cdot|$ also has a total well-founded model.

We noted earlier that the definition assures that I is a supported model. We can even give a somewhat stronger property of this model, as the following lemma will show. Before we can state the lemma resp. prove it, we need the notion of a well-supported interpretation [Fag91]:

3.18 Definition An interpretation I of a program P is called *well-supported* if there exists a strict well-founded ordering $\prec$ on I such that for any atom $A \in I$ there exists a clause $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m \in \text{ground}(P)$ such that $I \models A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$ and $A_i \prec A$ for each $i = 1, \dots, n$.

By a result of Fages in [Fag91, Theorem 2.1] the well-supported models of P are exactly the stable models of P. This means, that the notion of well-supportedness captures the stability of a model in a semi-syntactical manner that is somewhat akin to the level mapping approach. We will make use of this equivalence in the proof of the following lemma.

3.19 Lemma If a program P is well-founded w.r.t. a model I and a level mapping $|\cdot|$, then the program has a stable model, in fact, I is a stable model.

Proof: Condition 1 of Definition 3.13 implies that for any atom $A \in I$ there exists a clause $A \leftarrow \text{body} \in \text{ground}(P)$ such that $I \models \text{body}$ and $|A| > |A_i|$ for all $A_i \in \text{body}^+$. Since the level mapping induces a strict well-founded partial ordering on I, we have that I is a well-supported interpretation, and thus I is a stable model. ■

The existence of a stable model is necessary but by far not sufficient for a program to have a total well-founded model. We will now do the main work, given by the following proposition.

3.20 Proposition If a program P is well-founded w.r.t. a model I and a level mapping $|\cdot|$ then it has a total well-founded model, in fact, I is the total well-founded model.

Proof: From Lemma 3.19 we know, that I is a stable model of P. We thus have, from Theorem 3.6, that I is a fixed point of $\mathbf{W}_P$ and, by Corollary 3.7, that the well-founded partial model is a subset of I. Hence there must exist an ordinal α, such that $\mathbf{W}_P \uparrow (\alpha+1) = \mathbf{W}_P \uparrow \alpha \subseteq I$ is the least fixed point of $\mathbf{W}_P$. We now show, that this least fixed point is total. For this assume that α is the program's closure ordinal, i.e. that $\mathbf{W}_P \uparrow \alpha = \mathbf{W}_P \uparrow (\alpha + 1)$, and that β is the least ordinal, such that there exists an atom A with $|A| = \beta$ and neither $A \in \mathbf{W}_P \uparrow \alpha$ nor $\neg A \in \mathbf{W}_P \uparrow \alpha$. We consider the following two cases:

(1) If $I \models A$, we have that there exists a clause $A \leftarrow \text{body} \in \text{ground}(P)$ such that $I \models \text{body}$ and $|L| < |A|$ for all $L \in \text{body}$. Since we have that $\mathbf{W}_P \uparrow \alpha \subseteq I$ and for all atoms B with level less than β we have either $B \in \mathbf{W}_P \uparrow \alpha$ or $\neg B \in \mathbf{W}_P \uparrow \alpha$ we have that $\mathbf{W}_P \uparrow \alpha \models L$ for all $L \in \text{body}$ and thus $A \in \mathbf{T}_P(\mathbf{W}_P \uparrow \alpha)$, hence $A \in \mathbf{W}_P \uparrow (\alpha + 1)$ yielding a contradiction.

(2) If $I \not\models A$ we have that for all clauses of the form $A \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m \in \text{ground}(P)$ there exists a positive body literal

A_i such that $I \not\models A_i$ and $|A| \geq |A_i|$ or there exists a negative body literal $\neg B_j$ such that $I \models B_j$ and $|A| > |B_j|$. The latter then implies by induction hypothesis that $\neg B \in \mathbf{W}_P \uparrow \alpha$. Thus there must exist a clause satisfying the first condition, the literal(s) satisfying the condition being undefined in $\mathbf{W}_P \uparrow \alpha$, otherwise $A \in \mathbf{U}_P(\mathbf{W}_P \uparrow \alpha)$ yields a contradiction. In this clause we also must have $|A| = |A_i|$, by induction hypothesis.

Now consider the set S containing A and further all atoms B such that $|A| = |B|$, $I \not\models B$ and neither $B \in \mathbf{W}_P \uparrow \alpha$ nor $\neg B \in \mathbf{W}_P \uparrow \alpha$, i.e. S contains all atoms with the same level as A, that are false in I but undefined in $\mathbf{W}_P \uparrow \alpha$. We show that S is an unfounded set w.r.t. I, which then concludes the proof, since $S \subseteq \mathbf{U}_P(\mathbf{W}_P \uparrow \alpha)$, hence $\neg A \in \mathbf{W}_P \uparrow (\alpha + 1)$, which is a contradiction.

From [GRS91, Lemma 3.5], we know, that no clause in ground(P) is weakly falsified by $\mathbf{W}_P \uparrow \alpha$. We now show that no clause in ground(P) is weakly falsified by $\mathbf{W}_P \uparrow \alpha \cup \neg \cdot S$ either. For this, let $B \leftarrow \text{body} \in \text{ground}(P)$ be a clause such that $B \in S$. From $B \in S$, we obtain that $I \not\models B$, hence (because P is well-founded w.r.t. I and $|\cdot|$) $I \models \neg B$ and there must exist some atom $L_i \in \text{body}$ (resp. $\neg L_i \in \text{body}$) such that $I \not\models L_i$ (resp. $I \models L_i$). If $|B| > |L_i|$ then we have $\neg L_i \in \mathbf{W}_P \uparrow \alpha$ (resp. $L_i \in \mathbf{W}_P \uparrow \alpha$), and thus $B \leftarrow \text{body}$ is not weakly falsified by $\mathbf{W}_P \uparrow \alpha \cup \neg \cdot S$. Otherwise we must have a positive body literal L_i such that $I \not\models L_i$ and $|B| = |L_i|$, and from this either $\neg L_i \in \mathbf{W}_P \uparrow \alpha \subseteq I$ or $L_i \in S$, since $L_i \in S$ only if $\neg L_i \notin \mathbf{W}_P \uparrow \alpha$. But from this also follows, that $B \leftarrow \text{body}$ is not weakly falsified by $\mathbf{W}_P \uparrow \alpha \cup \neg \cdot S$. Thus no clause $B \leftarrow \text{body} \in P$ can be weakly falsified by $\mathbf{W}_P \uparrow \alpha \cup \neg \cdot S$. But this, from Lemma 3.3, implies, that S is an unfounded set w.r.t. $\mathbf{W}_P \uparrow \alpha$ (more formal, $S \cup (\mathbf{W}_P \uparrow \alpha)^-$ is an unfounded set w.r.t. $\mathbf{W}_P \uparrow \alpha^+$, where $\mathbf{W}_P \uparrow \alpha^- \subseteq \mathbf{W}_P \uparrow \alpha$ denotes the set that contains only the negative literals in $\mathbf{W}_P \uparrow \alpha$, whereas $\mathbf{W}_P \uparrow \alpha^+$ is the set that contains the positive literals). Thus the unfounded set S must be a subset of the greatest unfounded set $\mathbf{U}_P(\mathbf{W}_P \uparrow \alpha)$, and therefore $\neg A \in \mathbf{W} \uparrow (\alpha + 1)$, yielding a contradiction to the assumption that $\mathbf{W}_P \uparrow \alpha$ is the least fixed point of $\mathbf{W}_P$.

Thus we have that $\mathbf{W}_P \uparrow \alpha$ must be total for some α. Since $\mathbf{W}_P \uparrow \alpha \subseteq I$ for all α, we have that I is the total well-founded model, hence I is also the unique stable model of P. ∎

Having now finished the converse direction, we are ready to state the equality of both definitions. This is done in the following theorem.

3.21 Theorem (Main Theorem 1) A program P is well-founded w.r.t. a model I and a level mapping $|\cdot|$ if and only if P has a total well-founded model. Moreover, I is the total well-founded model of P.

Proof: Immediate from Propositions 3.16 and 3.20. ∎

The consideration of the effectively stratifiable programs is now especially easy, since we have established the equality of our characterization with the more general notion of programs with total well-founded model. We just need to restrict ourselves to the special case of finite program instantiations resp. finite propositional programs and easily obtain the following corollary:

3.22 Corollary A finite propositional program P is effectively stratifiable if and only if it is well-founded w.r.t. a model I and a level mapping $|\cdot|$.

Proof: From Theorem 3.12, we know that a finite propositional program is effectively stratifiable if and only if it has a total well-founded model. Furthermore, Theorem 3.21 assures that a program has a total well-founded model if and only if it is well-founded w.r.t. some model I and some level mapping $|\cdot|$. Thus the proof is complete. ■

Now that Theorem 3.21 has stated the equality of both definitions, we may as well abandon the term "well-founded w.r.t. a model I and a level mapping $|\cdot|$", because we now know that Definition 3.13 yields exactly the programs, that have a total well-founded model.

4 A new characterization of weakly stratified programs

We will now proceed with the second class of programs we want to consider resp. our new characterization of this class. However, both the characterization and the equivalence proof will not be as easy as those in the previous section, due to the definition of the weakly stratified programs, that relies heavily on a sophisticated program transformation. Thus we will have to develop a vast machinery of lemmata to establish the equality. As we will discuss in more detail later on, this complicated definition has caused some erroneous assumptions about the nature of the weakly stratified programs in the literature. We hope, that our new characterization will provide a convincing rectification of the common view of weakly stratified programs in the sense that level mapping based definitions are much more intuitive than complicated program transformations.

The section will begin with a detailed repetition of the preliminaries and notions needed for the definition of the weakly stratified programs and then proceed with stating our new characterization resp. the equivalence proof.

4.1 Preliminaries for weakly stratified programs

In this subsection we give the definitions of (partial) weakly perfect models that were introduced in [PP90]. In the following we will use $<$ and $\leq$ to be the dependency relations on ground atoms, see Section 2. Since the definitions in this paper were given for ground instatiations of programs, we will assume in the present section, that P is the ground instantiation of some first-order program.

The first considerations are mainly concerned with the atom dependency graph of the program, since certain properties of this graph are used in the program transformation later on. The first thing, we need to define, is an equivalence relation on ground atoms, which can be found in [PP90, Definition 3.1].

4.1 Definition (Equivalences on ground atoms) Let $\sim$ be the equivalence relation between ground atoms of P defined as follows:

$$A \sim B \equiv (A = B) \vee (A < B \wedge B < A)$$

We will call its equivalence classes *components* of the program's dependency graph G_P. A component is *trivial*, if it consists of a single element A such that $A \not< A$.

We now repeat an ordering relation on the components of the dependency graph G_P, due to [PP90, Definition 3.2]:

4.2 Definition Let C_1 and C_2 be two components of G_P. We define

$$C_1 \prec C_2 \equiv C_1 \neq C_2 \wedge \forall A_1 \in C_1, \forall A_2 \in C_2 (A_1 < A_2)$$

We call a component C_1 *minimal*, if there is no component C_2 such that $C_2 \prec C_1$.

Now we can define the two main notions in the definition of the weakly stratified programs, the bottom stratum and the bottom layer, as given in [PP90, Definition 3.4]:

4.3 Definition (Bottom stratum resp. bottom layer) By the *bottom stratum* $S(P)$ of P we mean the union of all minimal components of P, i.e.

$$S(P) = \bigcup \{C \ : \ C \text{ is a minimal component of } G_P\}$$

By the *bottom layer* $L(P)$ we mean the corresponding subprogram of P, i.e. $L(P)$ is the set of all clauses from P, whose heads belong to $S(P)$. Herbrand models of the subprogram $L(P)$ will be identified with signed subsets of the bottom stratum $S(P)$.

Using these notions, the reduction modulo a model, [PP90, Definition 3.5], is defined as follows:

4.4 Definition (Reduction modulo a model) Let P be a logic program and M be a partial interpretation of P. By a *reduction of P modulo M* we mean a new program $\frac{P}{M}$ obtained from P by performing the following two reductions:

- Remove from P all clauses which contain a body literal L such that $M \models \neg L$ or whose head belongs to M (in other words removing all clauses true in M).
- Remove from all the remaining clauses those body literals L which are satisfied in M, i.e. such that $M \models L$.

Finally, we also remove from the resulting program all non-unit clauses, whose heads appear as unit clauses also. This step ensures that the set of predicates appearing in unit clauses is disjoint from the set of predicates appearing in heads of non-unit clauses.

Now the (partial) weakly perfect models are transfinitely defined as follows [PP90, Definition 3.6]:

4.5 Definition (Weakly perfect model) Suppose P is a logic program and let $P_0 = P, M_0 = \emptyset$. Suppose that $\alpha > 0$ is a countable ordinal such that programs P_δ and partial interpretations M_δ have already been defined for all $\delta < \alpha$. Let

$$N_\alpha = \bigcup_{0<\delta<\alpha} M_\delta$$

$$P_\alpha = \frac{P}{N_\alpha}, S_\alpha = S(P_\alpha), L_\alpha = L(P_\alpha)$$

- If the program P_α is empty, then the construction stops and $M_P = N_\alpha$ is the *weakly perfect model* of P.
- Otherwise, if the bottom stratum S_α is empty or if the least model of the bottom layer L_α does not exist, then the construction also stops and $M_P = N_\alpha$ is the *partial weakly perfect model* of P.
- Otherwise, the partial interpretation M_α is defined as the least Herbrand model of the bottom layer L_α and the construction continues.

In the first two cases, α is called the *level* of P and is denoted by $\delta(P)$. For $0 < \alpha < \delta(P)$, the set S_α is called the α-th stratum of P and the program L_α is called the α-th layer of P.

In the process of constructing the strata S_α, some ground atoms may be eliminiated by the reduction and not fall into any stratum. Such atoms should be added to an arbitrary stratum, e.g. the first, and assumed false in M_P.

With the help of this construction, we define the notion of a weakly stratified program as follows [PP90, Definition 3.7]:

4.6 Definition We say that a logic program P is *weakly stratified* if it has a weakly perfect model and if all of its strata S_α consist only of trivial components or – equivalently – if all of its layers L_α are positive logic programs. In this case, we call the set of the program's strata $\{S_\alpha : 0 < \alpha < \delta(P)\}$ the weak stratification of P.

Note that in Definition 4.5, it is just required that the bottom layer has a least Herbrand model, which is then used to continue the reduction. However, we are not aware of a level mapping based characterization (or at least any other convincing resp. intuitive characterization) of the least Herbrand model of a general logic program. This situation prevents the characterization of the weakly perfect model semantics in terms of level mappings. The restriction to definite layers in Definition 4.6 however simplifies the case tremendously, in that the least Herbrand model for definite programs is easily given by the least fixed point of the monotonic and continuous operator T_P, which by the Kleene fixed point theorem is $T_P \uparrow \omega$.

Note also, that there was some confusion in the literature about the above equivalence. In [BF91], Remark 5.3, it was mentioned that this equivalence does not hold and therefore the definition of weakly-stratified programs is ambiguous. It was proposed to define the class of weakly-stratified-a programs to be those, that have all strata trivial, and the class of weakly-stratified-b programs as those, that have all layers definite. An unfortunate side-effect of this was that the weakly-stratified-a programs did not fit in the hierarchy of logic programs any longer, since there exist programs that are weakly-stratified-a, but have no well-founded model, as in [BF91], Remark 5.3, but there are also programs that have a well-founded model and are not weakly-stratified-a, like in [BF91], Example 5.6. However, this non-equivalence is due to a difference in the definition of trivial components in [BF91]. There, trivial components were defined to be components consisting of a single element, which is a weaker condition than in Definition 4.1, where it is also required that $A \not< A$ holds for the elements in the minimal component. This little difference causes A in the program $A \leftarrow \neg A$ to be a minimal, trivial component, according to the Definition in [BF91]. Since the bottom stratum contains A, whereas the bottom layer is not a definite program, the definition is ambiguous. If we take the "full" definition (Definition 4.1), we have that the bottom stratum of this program is empty. In general, an atom can not be in the bottom stratum of a program P, if there exists a clause $A \leftarrow \text{body} \in P$ with $\neg A \in \text{body}$, since in this case $A > A$ holds. Thus the definition of trivial components is not ambiguous, if we take $A \not< A$ as necessary condition for an atom A to be in the bottom stratum.

4.2 An alternative definition of weakly stratified programs

In the following, we will state our new characterization of weakly stratified programs. As the coincidence of our characterization and the original definition in [PP90] is not obvious and subject to an extensive proof, we will introduce the new (and temporary) notion of a program, that is *weakly stratified w.r.t. a model I and a level mapping* $|\cdot|$, following the conventions of the previous chapter. The only purpose of this temporary notion is to distinguish between the two different definitions, as long as they are not proven to be equivalent. The definition is now as follows:

4.7 Definition A program P is *weakly stratified w.r.t. a model I and a level mapping* $|\cdot|$ if and only if there exists a model I and a level mapping $|\cdot| : \mathcal{B}_P \to \gamma$ such that for every atom $A \in \mathcal{B}_P$ either one of the following conditions holds:

1. There exists a clause $A \leftarrow L_1, \ldots, L_n \in \text{ground}(P)$ such that $I \models L_1, \ldots, L_n$ and $|A| > |L_i|$ for all $i = 1, \ldots, n$.
2. $I \not\models A$ and for each clause $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m \in \text{ground}(P)$ (at least) one of the following three conditions holds:
   (a) There exists $i \in 1, \ldots, n$ such that $I \not\models A_i$ and $|A| > |A_i|$.

(b) There exists $i \in 1, \ldots, n$ such that $I \not\models A_i$, for all $i = 1, \ldots, n$ we have $|A| \geq |A_i|$ and for all $j = 1, \ldots, m$ we have $|A| > |B_j|$.
(c) There exists $j \in 1, \ldots, n$ such that $I \models B_j$ and $|A| > |B_j|$.

Note that this definition implies that I is a supported model. If for an atom $A \in \mathcal{B}_P$ we have $I \models A$ we also have by condition 1 of the above definition that there exists a clause $A \leftarrow$ body with $I \models$ body. Otherwise we have that $I \not\models A$. Thus the model is supported.

Before we can prove the first part of the equivalence, that every weakly stratified program is weakly stratified w.r.t. a model I and a level mapping $|\cdot|$, we need to fix some preliminaries. As in the previous section, we will indeed show something stronger. In fact, we will show, that the program is weakly stratified w.r.t. the unique weakly perfect model and a canonical level mapping that we will define below. However, the situation is a little more complex. This is due to the program transformation, which essentially consists of two parts, i.e. we have the weak stratification, and each of the layers corresponding to the program's strata, we have a (tractable) least Herbrand model. We will thus map the elements of $\mathcal{B}_P$ to a pair of ordinals instead of a single one. We will argue later on, that this does not affect our definition, and is only a helpful construction.

Now, for these pairs of ordinals we will use the common lexicographic ordering, i.e.

$$(\gamma_1, \delta_1) < (\gamma_2, \delta_2) \equiv (\gamma_1 < \gamma_2) \vee ((\gamma_1 = \gamma_2) \wedge (\delta_1 < \delta_2))$$

Thus in the following, the canonical level mapping will be denoted by $|\cdot|_P : \mathcal{B}_P \to (\gamma, \delta)$. Since we are using the construction of the weakly perfect model, we use the terminology of Definition 4.5.

4.8 Construction (Canonical level mapping) Let P be a weakly stratified program and S_α be the bottom stratum of the program P_α. For every atom $A \in S_\alpha$ we set $|A|_P = (\alpha, \beta)$ with β taken to be the least ordinal such that $A \in T_{L_\alpha} \uparrow \beta$ if $M_\alpha \models A$, and ω otherwise.

Note that since we consider weakly stratified programs, i.e. all layers are definite programs, for the second ordinal in the level $|A|_P = (\alpha, \beta)$ we always have $\beta \leq \omega$, which will be of use later on.

Thus the first ordinal in the level (α, β) of A denotes that A is in the α-th stratum S_α and β denotes the level of A w.r.t. T_{L_α} that describes the semantics of the α-th layer L_α. Note that in our construction we suppose the least Herbrand model of L_α to be constructed by

$$M_\alpha = T_{L_\alpha} \uparrow \omega \cup \neg \cdot (S_\alpha \setminus T_{L_\alpha} \uparrow \omega)$$

We express the least Herbrand model M_α as three-valued interpretation, in order to fit the construction from Definition 4.5, since there partial interpretations are required. We thus need to explicitly include the negative literals.

This is due to M_α being a total interpretation for S_α, but usually only a partial interpretation for P.

This level mapping satisfies the following immediate conditions, giving some technical details we will need later on:

4.9 Lemma Let P be a weakly stratified program and $|\cdot|_P$ be its canonical level mapping.

1. If $|A|_P = |B|_P$ then there exists an ordinal α such that $A, B \in S_\alpha$.
2. If $A \in S_\alpha, B \in S_\beta, \beta > \alpha$ then $|A|_P < |B|_P$.

Proof: (1) Immediate, since if $|A|_P = |B|_P = (\alpha, \beta)$ we have $A, B \in S_\alpha$.

(2) Also immediate for $A \in S_\alpha, B \in S_\beta, \alpha < \beta$ implies $|A|_P = (\alpha, \epsilon_1) < (\beta, \epsilon_2) = |B|_P$ for some ϵ_1, ϵ_2 because of the lexicographic ordering. ■

It is an important condition for the applicability of the canonical level mapping in Definition 4.7 to be total, which is guaranteed by the following lemma.

4.10 Lemma Let P be a weakly stratified program. The level mapping $|\cdot|_P$ obtained from Construction 4.8 is total, i.e. for every atom $A \in \mathcal{B}_P$ there exist ordinals α, β such that $|A|_P = (\alpha, \beta)$.

Proof: P is weakly stratified, therefore by Definition 4.6 it must have a weakly perfect model. Thus by Definition 4.5 every atom must be member of some stratum S_α and be true or false in M_α. If $M_\alpha \models A$ then there must exist an ordinal β such that $A \in T_{L_\alpha} \uparrow \beta$, in this case we have $|A|_P = (\alpha, \beta)$, or $M_\alpha \not\models A$, for which we have $|A|_P = (\alpha, \omega)$. ■

The last prerequisite we need is the notion of the restriction of a program, which is defined as follows:

4.11 Definition Let P be a weakly stratified program and $S_\alpha \subseteq \mathcal{B}_P$ be a stratum of P. The *restriction of* P *to* S_α is the program $P' \subseteq P$ containing only those clauses with head $A \in S_\delta$ with $\delta \leq \alpha$. We will denote this by $P|_{S_\alpha}$. Note that in fact $P|_{S_\alpha} = \bigcup_{\delta \leq \alpha} L_\alpha$.

For our proof, we will also need a subsumption relation in order to formalize the notion of what happens during the reduction, and what are the relationships of the programs before and after the reduction modulo a model. Since the subsumption relation from Definition 3.10 does not suit our purposes, we will modify it slightly. Moreover, we add a certain semantic aspect to it, as can be seen in the following.

4.12 Definition (Model-consistent subsumption relation) Let r and s be two clauses, P and Q be two logic programs and let I be an interpretation. Then

1. $r \preceq s$ if and only if head$(r) =$ head(s) and body$(r) \subseteq$ body(s). We say r *subsumes* s.

2. $P \preceq Q$ if and only if for each clause $r \in P$ there exists a clause $s \in Q$ such that $r \preccurlyeq s$. We say P *subsumes* Q.

3. We say that P *subsumes* Q *model-consistently (w.r.t.* I*)*, denoted by $P \preceq_I Q$, if and only if the following two conditions hold:

   i) For each clause $r \in P$ there exists a clause $s \in Q$ such that $r \preceq s$ and $I \models \text{body}(s) \setminus \text{body}(r)$.

   ii) For each clause $s \in Q$ with $I \models \text{body}(s)$ and head$(s) \notin I$ there exists a clause $r \in P$ such that $r \preceq s$.

The purpose of this definition is, that the reduction modulo a model is in the above sense model-consistent, and further, model-consistent subsumption has some advantages when handling level mappings. We will formalize these thoughts in the following lemma and make use of this in our equivalence proofs.

4.13 Lemma Let P be a weakly stratifiable program, α be an ordinal and $N_\alpha = \bigcup_{0<\delta<\alpha} M_\delta$ be as in Definition 4.5. We then have, that P reduced modulo N_α subsumes P consistent with N_α, i.e.

$$\frac{P}{N_\alpha} \preccurlyeq_{N_\alpha} P.$$

Proof: For each clause $c \in \frac{P}{N_\alpha}$ there must exist a clause $d \in P$ such that $c \preccurlyeq d$, this is immediate from Definition 4.4, since $\frac{P}{N_\alpha}$ is derived from P by reduction modulo N_α. Now further, since literals are removed from clauses, if they are true under the model, we immediately have $N_\alpha \models \text{body}(d) \setminus \text{body}(c)$. Now for every clause in $A \leftarrow \text{body} \in P$ with $N_\alpha \models \text{body}$ and whose head is not true in N_α, we have the unit clause $A \leftarrow$ in $\frac{P}{N_\alpha}$, since this clause can not be removed according to the definition of reduction modulo a model, Definition 4.4. Thus we have $\frac{P}{N_\alpha} \preccurlyeq_{N_\alpha} P$. ■

Now that the main tool for the proof, the model consistent subsumption, is established, we will prove the first part of the equivalence. However, there are still some conventions to adjust: For the proof, we do not assume all eliminated atoms to be collected in the first stratum, as it was the case in Definition 4.5. We will instead adopt the following convention: If α is the least ordinal, such that the atom A is contained in all P_δ for all $\delta < \alpha$, but not in P_α, we assume A to be in the stratum S_α. This is no loss of generality, since it was emphasized in Definition 4.5, that these atoms can be considered to be contained in an arbitrary stratum.

4.14 Proposition If P is a weakly stratified program, then it is also weakly stratified w.r.t. a model I and a level mapping $|\cdot|$, where I is the unique weakly perfect model of P and $|\cdot|$ is the canonical level mapping $|\cdot|_P$ from Construction 4.8.

Proof: We will do the proof by transfinite induction. Since we are using pairs of ordinals and not ordinals, this is a little more complicated. We therefore first discuss the cases to be considered: Let $|A|_P = (\alpha, \delta)$. The overall construction, reflected by the first ordinal, is a transfinite sequence (Definition 4.5) of finding strata, finding least models of layers and reducing modulo them. We clearly have to do a transfinite induction on the number of strata. From the structure of Definition 4.5, we see, that successor and limit case are to be treated similarly, since the induction hypothesis in step α is in both cases given for all ordinals $\beta < \alpha$.

The proof of this step itself – finding the least Herbrand model of the bottom layer – is done by induction on the powers of T_{L_α}. We only have to consider the base and the successor case, since L_α is definite for any α.

The overall proof structure is now the following: We assume that the conditions of Definition 4.7 holds for all atoms A with $|A|_P = (\delta, \epsilon)$ and $\delta < \alpha$. Now we prove, that if L_α is weakly stratified w.r.t. I and $|\cdot|_P$, then so is $P|_{S_\alpha}$. Then we prove by induction on the powers of T_{L_α}, that L_α is in fact weakly stratified w.r.t. I and $|\cdot|_P$. Thus, by modus ponens, we conclude that $P|_{S_\alpha}$ is weakly stratified w.r.t. I and $|\cdot|_P$. Then, by transfinite induction on the strata ordinals, we have that $P = P|_{S_{\delta(P)}}$ is itself weakly stratified w.r.t. I and $|\cdot|_P$, $\delta(P)$ denoting the level of the program.

Note, that in addition to this, we will need the fact, that the sequence of partial models as in Definition 4.5 approximates the unique weakly perfect model. But this follows trivially from the assumption, that I is the unique weakly perfect model, entailing $N_\alpha \subseteq I$ for all α.

In the following let P be a weakly stratified program, I its weakly perfect model, $|\cdot|_P$ the canonical level mapping of P and $\alpha > 0$ a countable ordinal.

The induction hypothesis is now, that for all atoms $A \in \mathcal{B}_P$ with level (δ, ϵ) with $\delta < \alpha$ the conditions of Definition 4.7 hold, i.e. the programs $P|_{S_\delta}$ are weakly stratified w.r.t. I and $|\cdot|_P$.

We are now going to prove that the proposition holds for all atoms $A \in \mathcal{B}_P$ from the stratum S_α. We now consider the two cases noted above in the outline of the proof structure:

(1) We show, that if L_α is weakly stratified w.r.t. I and $|\cdot|_P$, then $P|_{S_\alpha}$ is also weakly stratified w.r.t. I and $|\cdot|_P$.

In order to do this, we first show, that $L_\alpha \preceq_{N_\alpha} P|_{S_\alpha}$, recall Definition 4.13.

L_α is the subprogram of $\frac{P}{N_\alpha}$ that contains all clauses with head in S_α. These clauses are also contained in $\frac{P|_{S_\alpha}}{N_\alpha}$, since this restriction of P to S_α

still contains those clauses with head in S_α. We therefore have $L_\alpha \subseteq \frac{P|_{S_\alpha}}{N_\alpha}$. Since for every clause $c \in L_\alpha$ there exists a clause $d \in P|_{S_\alpha}$ such that c was obtained by reduction modulo $N_\alpha = \bigcup_{0<\delta<\alpha} M_\delta$ from d (this entails $\text{body}(c) \subseteq \text{body}(d)$ and thus $c \preceq d$) we have that $L_\alpha \preceq P|_{S_\alpha}$. Moreover, for these clauses c and d we have that $N_\alpha \models \text{body}(d) \setminus \text{body}(c)$ and we have for each clause $d \in P|_{S_\alpha}$ with $N_\alpha \models \text{body}(d)$ and $N_\alpha \not\models \text{head}(d)$ there exists a clause $c \in L_\alpha$ with $c \preceq d$. We thus have $L_\alpha \preceq_{N_\alpha} P|_{S_\alpha}$. Now assume that L_α is weakly stratified w.r.t. I and $|\cdot|_P$. From the assumption we know that $P|_{S_\delta}$ is weakly stratified w.r.t. I and $|\cdot|_P$ for all $\delta < \alpha$. We show that either one of the conditions from Definition 4.7 holds for each $A \in S_\alpha$. We distinguish two cases:

(a) If $I \models A$ we have that there exists a clause $A \leftarrow A_1, \ldots, A_n$ in L_α such that $I \models A_1, \ldots, A_n$ and $|A|_P > |A_i|_P$ for all $i = 1, \ldots, n$.

From $L_\alpha \preceq_{N_\alpha} P|_{S_\alpha}$ we know that there must exist a clause $A \leftarrow L_1, \ldots, L_m$ in $P|_{S_\alpha}$ with $I \models L_1, \ldots, L_m$ and $\{A_1, \ldots, A_n\} \subseteq \{L_1, \ldots, L_m\}$. All literals $L \in L_1, \ldots, L_m$ with $L \notin A_1, \ldots, A_n$ have been reduced modulo N_α and thus we have $L \in S_\delta$ for some $\delta < \alpha$ and by Lemma 4.9 we have $|L|_P < |A|_P$. We therefore have $|A|_P > |L_i|_P$ for all $i = 1, \ldots, m$ and the condition 1 of Definition 4.7 holds. If otherwise all clauses were removed by the reduction modulo N_α, then $N_\alpha \models A$, thus we have $A \in S_\delta$ for some $\delta < \alpha$, and by induction hypothesis, we have that Condition 1 is satisfied for the atom A.

(b) If otherwise $I \not\models A$ we have that for all clauses $A \leftarrow A_1, \ldots, A_n$ in L_α there exists $i \in 1, \ldots, n$ such that $I \not\models A_i$ and one of the conditions 2.a-c from Definition 4.7 holds.

Now assume $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m \in P|_{S_\alpha}$. (If no such clause exists, the Condition 2 of Definition 4.7 is satisfied trivially.) If there exists a clause $A \leftarrow \text{body} \in L_\alpha$ such that $\text{body} \subseteq \{A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m\}$ and conditions 2.a or 2.c are satisfied for some literal $L \in \text{body}$, then the corresponding condition is also satisfied for $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m \in P|_{S_\alpha}$. If otherwise Condition 2.b is satisfied for $A \leftarrow \text{body} \in L_\alpha$, we know, that there must not be negative literals contained in body. Thus $\neg B_1, \ldots, \neg B_m$ have been reduced modulo N_α, i.e. $B_1, \ldots, B_m \in S_\delta$ for some $\delta < \alpha$. The same also holds for positive atoms, that were removed. Now for all $A_i \in \text{body}$ we have $|A|_P \geq |A_i|_P$, since L_α is weakly stratified w.r.t I and $|\cdot|_P$. Thus Condition 2.b is satisfied for $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m \in P|_{S_\alpha}$, since there exists $i = 1, \ldots, n$ such that $A_i \in \text{body}$ and $I \not\models A_i$. Further, for all $j = 1, \ldots, n$ we have $|A|_P \geq |A_j|_P$ and for all $k = 1, \ldots, m$, by Lemma 4.9, we have $|A|_P > |B_k|_P$.

If now otherwise there does not exist a clause in L_α that subsumes $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m \in P|_{S_\alpha}$, its body must contain a literal L such that $I \not\models L$ and $\neg L \in N_\alpha$, because the clause was reduced modulo N_α. We therefore have $L \in S_\delta$ for some $\delta < \alpha$, hence either Condition 2.a is satisfied

– if L is a positive literal – or Condition 2.c is satisfied – if L is a negative literal.

Note that the atoms that do not fall into any stratum according to Definition 4.5 and that are contained in the stratum S_α according to our new convention, are reduced modulo N_α. Thus, if A is such an atom, then in every clause with head A there must be a literal L with $I \not\models A$ and $L \in S_\delta$ for some $\delta < \alpha$ and by the same argument as above, we have that condition 2 of Definition 4.7 is satisfied for these literals as well.

We now have, from the assumption, that if L_α is weakly stratified w.r.t. I and $|\cdot|_P$, then so is $P|_{S_\alpha}$.

(2) We show that L_α is weakly stratified w.r.t. I and $|\cdot|_P$.

From Construction 4.8 we know that for all atoms $A \in S_\alpha$ with $I \not\models A$ we have $|A|_P = (\alpha, \omega)$. Since there exist no negative body literals in L_α, we trivially have, that condition 2.b from Definition 4.7 is satisfied for all A in S_α with $\neg A \in N_\alpha$. We now consider the atoms A with $I \models A$. This is done by induction on the powers of T_{L_α}. Therefore assume that the conditions of Definition 4.7 hold for all atoms $A \in \mathcal{B}_P$ with $|A|_P < (\alpha, 0)$.

We consider only the successor case, since the base case is trivial, because there does not exist an atom $A \in \mathcal{B}_P$ such that $A \in T_{L_\alpha} \uparrow 0$. Now assume that for all atoms A with level less than $(\alpha, \beta + 1)$ the conditions from Definition 4.7 are satisfied. We show they are satisfied for the atoms with level $(\alpha, \beta + 1)$, too.

If $\beta + 1$ is the least ordinal such that $A \in T_{L_\alpha} \uparrow (\beta + 1)$, there must exist a clause $A \leftarrow A_1, \ldots, A_n \in L_\alpha$ such that for all A_i we have that there exists $\delta < \beta$ such that $A_i \in T_{L_\alpha} \uparrow \delta$. We thus have $I \models A_1, \ldots, A_n$ and $|A|_P > |A_i|_P$ for all $i \in 1, \ldots, n$. Thus, by induction, we have that L_α is weakly stratified or - equivalently - all atoms from S_α satisfy the conditions from Definition 4.7.

Now from (1) and (2) we have by transfinite induction on the strata ordinals of P that P is in fact weakly stratifiable w.r.t. the unique weakly perfect model I and the canonical level mapping $|\cdot|_P$. Thus the proof is complete. ■

Note that there are no complications with the level mapping. Since for the second ordinal in (α, β) we have $\beta \leq \omega$, the pair of ordinals (α, β) represents a well-founded ordering and is therefore order-isomorphic to some ordinal γ.

We are now going to show the converse direction, i.e. that if a program is weakly stratified w.r.t. some model I and some level mapping $|\cdot|$, then it is also weakly stratified, as stated by the following proposition:

4.15 Proposition If a program P is weakly stratified w.r.t. a model I and a level mapping $|\cdot|$ then it is also weakly stratified, in fact I is the unique weakly perfect model of the program P.

Proof: The proof will be done by transfinite induction on the construction of the (partial) weakly perfect model. For this let P be a program that is weakly stratified w.r.t. a model I and a level mapping $|\cdot|$. We have to consider the following cases:

(1) We show that if P is non-empty, its bottom stratum is also non-empty and consists only of trivial components. We therefore use the skeleton of the proof of Theorem 8.2.2 in [Hit01]. Note that we are determining the bottom stratum of $P_1 = \frac{P}{\emptyset}$, according to Definition 4.5.

Assume there exists some component $C \subseteq S(P_1)$ which is not trivial. Then there must exist an atom $A \in C$ that is least w.r.t. $|\cdot|$, since $|\cdot|$ is well-founded, and there must exist some $B \in C$ with $A < B$ and $B < A$ [PP90, Definition 3.1]. Now from [PP90, Proposition 3.3] we have that if C is minimal there can not exist any A' in any component different from C with $A' \leq A$. From this we have that all atoms occuring in clauses with heads in C must also be in C, especially all atoms occuring in clauses with head A must be in C. Thus if $I \models A$, then there must exist a unit clause $A \leftarrow$ in P, otherwise we had a clause $A \leftarrow A_1, \ldots, A_n$ with $I \models A_1, \ldots, A_n$ and $|A| > |A_i|$ for all $i = 1, \ldots, n$, contradicting the minimality of A w.r.t. $|\cdot|$. Since the first step before determining the bottom stratum is a reduction modulo $\emptyset$ the only clause with head A in P_1 is $A \leftarrow$. Hence $\{A\}$ is a trivial component and can not be member of a non-trivial component. If otherwise we have $I \not\models A$ we can, by Definition 4.7, not have any clauses satisfying conditions 2.a or 2.c, since this would imply the existence of some $A' \leq A$ or some $B' < A$ entailing $|A'| < |A|$ resp. $|B'| < |A|$, which is a contradiction to the minimality of A w.r.t. $|\cdot|$. We therefore have, that every clause with head A must satisfy condition 2.b. Furthermore there may not be negative literals in the body, $(B' < A)$, i.e. for all literals in the bodies of the clauses with head A we have $|A_i| \leq |A|$ and in every clause there must be a literal A_j with $I \not\models A_j$. It follows also, that all these literals then must be contained in $C \subseteq S(P_1)$. Now take some literal $B \in C$ with $A > B$. From $A > B$ we have that there must exist a chain $A = A_0, A_1, A_2 \ldots A_n$ such that $I \not\models A_i$ and $|A| = |A_i|$ for all $i \in I$. Each of the clauses yielding the literals in this chain must satisfy condition 2.b. Now there must exist a clause $A_n \leftarrow \neg B, L_1, \ldots, L_n \in P_1$, otherwise we had $A \not> B$ which is a contradiction. But in this case we have $|A| = |A_n| > |B|$, yielding also a contradiction.[1] All components of the bottom stratum must therefore be trivial. Note that if there are no clauses with head A, A is also a trivial component in the bottom stratum.

We now show that the bottom stratum is non-empty. This holds, since $|\cdot|$ is a well-founded ordering, thus there must exist minimal atoms w.r.t. $|\cdot|$. Such a minimal atom A is, by the above considerations, either head of a unit clause, or part of a definite subprogram of P_1, or there do not exist any

[1]In fact, the clauses A depends on form a definite logic program.

clauses with head A, and thus A is minimal w.r.t. the dependency relation $<$. This implies, that the bottom stratum is non-empty.

(2) We show that the least Herbrand model of the bottom layer $M_{L(P_\alpha)}$ is consistent with I.

We do this by transfinite induction on the level of the atoms $A \in S(P_\alpha)$. Thus in the following assume that for all atoms $A \in S(P_\alpha)$ with level $\delta < \beta$, $M_{L(P_\alpha)}$ is consistent with I, i.e. $I \models A$ if and only if $M_{L(P_\alpha)} \models A$ and $I \not\models A$ if and only if $\neg A \in M_{L(P_\alpha)}$ resp. $M_{L(P_\alpha)} \subseteq I$. We show this holds also for atoms A with $|A| = \beta$.

Now if $M_{L(P_\alpha)} \models A$ we either have a unit clause $A \leftarrow$ in $L(P_\alpha)$, or we have a clause $A \leftarrow A_1, \ldots, A_n \in L(P_\alpha)$ with $M_{L(P_\alpha)} \models A_i$ and $|A| > |A_i|$ for all $i = 1, \ldots, n$, hence by induction assumption $I \models A_i$ for all $i = 1, \ldots, n$. In both cases there must exist a clause $A \leftarrow \text{body} \in P$ with $\{A_1, \ldots, A_n\} \subseteq \text{body}$ and $N_\alpha \models \text{body} \setminus \{A_1, \ldots, A_n\}$. We thus have $I \models A$.

If $I \models A$, we have a clause $A \leftarrow \text{body} \in P$ with $I \models \text{body}$ and $|A| > |L_i|$ for all $L_i \in \text{body}$. Since $N_\alpha \not\models A$, this clause is not removed modulo N_α, thus there must exist a clause $A \leftarrow A_1, \ldots, A_n \in L(P_\alpha)$ with $\{A_1, \ldots, A_n\} \subseteq \text{body}$. From our induction assumption, we have that I and $M_{L(P_\alpha)}$ are consistent for all atoms with level less than β, hence $I \models A_i$ implies $M_{L(P_\alpha)} \models A_i$ for all $i = 1, \ldots, n$ and thus $M_{L(P_\alpha)} \models A$.

Now since $M_{L(P_\alpha)}$ is a total model for $S(P_\alpha)$, we conclude that $\neg A \in M_{L(P_\alpha)}$ if and only if $I \not\models A$. Thus by transfinite induction on the level of the atoms in $S(P_\alpha)$ we have that $M_{L(P_\alpha)}$ is consistent with I.

(3) We show that the reduced program $\frac{P}{N_\alpha}$ is itself again weakly stratified w.r.t. I (but restricted to the atoms in $\mathcal{B}_P \setminus (\bigcup_{\beta<\alpha} S_\beta)$) and the level mapping $|\cdot|$. In abuse of notation we will denote the restricted model by $I|_{\mathcal{B}_P \setminus (\bigcup_{\beta<\alpha} S_\beta)}$

From Lemma 4.13, we know that $\frac{P}{N_\alpha} \preceq_{N_\alpha} P$. Now let $A \in \mathcal{B}_P \setminus (\bigcup_{\beta<\alpha} S_\beta)$. We distinguish two cases:

(a) If $I \models A$, then there must exist a clause $A \leftarrow \text{body} \in P$, since P is weakly stratified w.r.t. I and $|\cdot|$. Furthermore, for all $L \in \text{body}$ we have $I \models L$ and $|L| < |A|$. Now, since we can not have $N_\alpha \models A$, there must also exist a clause in $\frac{P}{N_\alpha}$, that subsumes $A \leftarrow \text{body}$. For this clause Condition 1 of Definition 4.7 is satisfied, by $\frac{P}{N_\alpha} \preceq_{N_\alpha} P$. Thus A satisfies Condition 1 of Definition 4.7.

(b) If otherwise $I \not\models A$, we must have for each clause $A \leftarrow \text{body}_{red} \in \frac{P}{N_\alpha}$ a subsumed clause $A \leftarrow \text{body} \in P$. Now there must exist some literal $L \in \text{body}_{red}$ such that $I \not\models L$, otherwise $I \models \text{body}$. Now by $\frac{P}{N_\alpha} \preceq_{N_\alpha} P$, we have that $N_\alpha \subseteq I \models \text{body} \setminus \text{body}_{red}$. Thus all literals not consistent with I, i.e. literals $L \in \text{body}$ with $\neg L \in I$ and $\neg L \notin N_\alpha$ are still contained in body_{red}. Note that there can not be a literal $L \in \text{body}$ with $\neg L \in N_\alpha$, otherwise the clause would be removed modulo N_α. Thus by the model consistency property, $A \leftarrow \text{body}_{red}$ must still satisfy the condition $A \leftarrow \text{body}$ met.

We therefore have that $\frac{P}{N_\alpha}$ is weakly stratified w.r.t. I and $|\cdot|$.

Note, that in this case, we do not need to distinguish successor and limit cases in the indices of the model N_α, since we have proven in Lemma 4.13, that the model consistency property holds for all ordinals α. It is only important, that N_α is a partial interpretation for P.

(4) Now via transfinite induction we have that every stratum S_α of P is non-empty (if P_α is non-empty) and the least Herbrand model of $L(P)$ does always exist and is consistent with I, thus we have, that P is weakly stratified and I is its unique weakly perfect model. ■

4.16 Theorem (Main Theorem 2) A program P is weakly stratified if and only if it is weakly stratified w.r.t. a model I and a level mapping $|\cdot|$. Moreover we have that I is the unique weakly perfect model of P.

Proof: This follows immediately from Propositions 4.14 and 4.15. ■

Now that Theorem 4.16 has stated the equality of both definitions, we may as well abandon the term "weakly stratifiable w.r.t. some model I and some level mapping $|\cdot|$", since we now know that Definition 4.7 yields exactly the weakly stratifiable programs.

5 Conclusion and Further Work

In the following section, the results obtained in the thesis will be reviewed from two points of view: First, we will use our new characterizations of the programs with total well-founded model resp. the weakly stratified programs to clarify their relationships with other program classes in the hierarchy of logic programs. This will yield an (informal) understanding of what program structures are permitted by the several program classes and therefore an intuitive comparison of the expressiveness of these program classes. The second part of this section will then be concerned with the question, how these results can be used resp. generalized to other logic program classes.

For the (informal) comparison of the program classes we dealt with throughout the sequel with other ones in the hierarchy of logic program classes, we will need the definition of two more program classes: the Φ-accessible programs [HS99] and the locally stratified programs [Prz88]. These have been – up to now and to the best of our knowledge – the maximal program classes in the hierarchy, that can be defined resp. understood with the help of level mappings. The definitions are now as follows:

A program P is called *Φ-accessible* if and only if there exists a level mapping $|\cdot|$ and a model I for P such that the following condition holds. Each $A \in \mathcal{B}_P$ satisfies either one of the following conditions:

(i) There exists a clause $A \leftarrow L_1, \ldots, L_n \in \text{ground}(P)$ with head A such that $I \models L_1, \ldots, L_n$ and $|A| > |L_i|$ for each $i = 1, \ldots, n$.

(ii) $I \not\models A$ and for each clause $A \leftarrow L_1, \ldots, L_n \in \text{ground}(P)$ with head A there exists $i = 1, \ldots, n$ such that $I \not\models L_i$ and $|A| > |L_i|$.

Note that this definition captures all programs, for which the Fitting model (obtained as the least fixed point of the operator Φ_P [Fit85]) is total.

A program P is called *locally stratified* if and only if there exists a level mapping $|\cdot|$ for P such that for every clause $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m \in \text{ground}(P)$ we have $|A| \geq |A_i|$ for all $i = 1, \ldots, n$ and $|A| > |B_j|$ for all $j = 1, \ldots, m$.

From these definitions the intuitive meaning and permitted expressiveness of the programs in the corresponding classes is easy to see: In locally stratified programs, there may occur recursion through positive atoms, since the condition for the positive atoms is only $|A| \geq |A_i|$. However, the restrictions on the atoms are imposed on every clause of the program. In the case of the Φ-accessible programs, this restriction is relaxed. For atoms that are true in the model I, only one clause must satisfy some restriction resp. for clauses with head false in I, only one literal in the body of the clause must satisfy the restriction. Nevertheless, the restrictions are stronger, in that we have $|A| > |L_i|$ for both positive and negative literals. The following two programs will clarify the relationship of both program classes:

5.1 Program

$$\begin{array}{rcl} A & \leftarrow & B. \\ B & \leftarrow & A. \end{array}$$

The level mapping $|\cdot|_1$ with $|A|_1 = |B|_1 = 0$ easily satisfies the conditions in the definition of locally stratified programs, hence Program 5.1 is locally stratified. However, for both models of the program, $\{A, B\}$ and $\emptyset$ there do not exist level mappings satisfying the conditions of the definition of Φ-accessible programs. Thus Program 5.1 is not Φ-accessible. The next program will be an example for a program that is Φ-accessible but not locally stratified.

5.2 Program

$$\begin{array}{rcl} A & \leftarrow & \neg B, A. \\ B & \leftarrow & \neg A. \\ B & \leftarrow & . \end{array}$$

It is easy to see, that there does not exist a level mapping for this program, that meets the conditions in the definition of locally stratified programs. However, every level mapping $|\cdot|_2$ with $|A|_2 > |B|_2$ will meet the conditions in the definition of Φ-accessible programs.

We thus have that every definite program is locally stratified, but may not be Φ-accessible. On the other hand we see, that the definition of Φ-accessibility permits recursion through negation in some way. This is due to the fact, that for Φ-accessibility the conditions need only hold for one clause, if the head is true in I, resp. for one literal in the clause, if the head is false in I.

Turning attention to the program classes we characterized in this work, we easily note, that every Φ-accessible program is both weakly stratified and has a total well-founded model. The attraction of our approach is now, that these two observations need no extensive proof, but can easily obtained by comparing the definitions – we easily see that in the definition of the weakly stratified programs resp. programs with total well-founded models the condition 2) is relaxed in different ways. Unfortunately, this observation can not be obtained that easy for the locally stratified programs, since for the locally stratified programs only a semi-strict level decrease is required for positive atoms. We thus need to refer to the proofs in the literature: In [PP90, Theorem 4.1] it was proven that every locally stratified program is weakly stratified, hence has a total well-founded model. This is due to the fact, that for the definition of weakly stratified programs and programs with total well-founded model, we need a strict decrease in the positive atoms in condition 1) to assure the existence of a stable model, see Lemma 3.19.

The definition of locally stratified programs does not depend on a model. Nevertheless we can always find a model, such that we have a strict decrease in condition 1) for the positive atoms, since every locally stratified program has a stable model.

We thus have, that both programs, 5.1 and 5.2, are weakly stratified and have a total well-founded model. Thus in these two program classes, recursion through positive atoms is permitted without any restrictions, whereas recursion through negation is controlled in some or the other way. The next program will show, where the difference in the handling of recursion through negation lies [BF91, Example 5.6]

5.3 Program

$$\begin{array}{rcl} A & \leftarrow & B. \\ B & \leftarrow & A. \\ C & \leftarrow & \neg A. \\ A & \leftarrow & B, \neg C. \end{array}$$

Clearly, this program has a total well-founded model: The level mapping $|\cdot|_3$ with $|A|_3 = |B|_3 = 0$ and $|C|_3 = 1$ satisfies the conditions from Definition 3.13 w.r.t. the model $\{\neg A, \neg B, C\}$. However, the conditions from Definition 4.7 are not satisfied for any level mapping. The problem lies in the last clause. $\{\neg A, \neg B, C\}$ is the unique stable model for Program 5.3. Assuming that Condition 2.a) is satisified for the last clause leads to a contradiction, since the second clause forbids $|A| > |B|$. The same problems occur with Conditions 2.b) and 2.c), $|A| > |C|$ can never be satisifed for the third clause of the program. This corresponds to the fact, that $\{A, B, C\}$ is the only (and non-trivial) component of the programs dependency graph, thus the bottom layer is not a definite program and thus Program 5.3 is not weakly stratified.

From a more abstract point of view, we can state, that there is a difference in the way the definitions of weakly stratified programs resp. programs with total well-founded models handle recursion through positive resp. negative atoms. This is due to the additional Condition 2.b) in Definition 4.7. The difference between the programs with total well-founded model and the weakly stratified programs is, that there are no constraints for recursion through positive atoms in the case of the programs having a total well-founded model, but for weakly stratified there is the constraint, that recursion through positive atoms may not be 'intertwined' with recursion through negative atoms in the atom dependency graph, i.e. an atom must not be part of both a positive and a negative cycle in the atom dependency graph. Note that some of the cycles in the dependency graph may be omitted by the reduction constructions for weakly stratified programs. This is also expressed by the Condition 2 in our semi-syntactical definitions, since there only one literal in the body must satisfy the required conditions and no restrictions are imposed on the other body literals. However, if a positive loop and a negative loop in an atom dependency graph are 'intertwined', this program is not weakly stratified, since in this case the bottom stratum is not trivial. Note that this is the case in program 5.3: We have a positive loop in the atom dependency graph through A and B, and a negative one through A and C. In Definition 4.7, this is captured by the Conditions 2.a and 2.b in Definition 4.7: If an atom A is not part of a positive cycle then we may choose some positive atom in the body to have a level lower than A. If A is part of a positive loop together with A both must have the same level, but in this case there may not be any negative cycle through A and this is expressed with the condition $|A| > |B_j|$ for all $B_j \in \mathrm{body}^-$.

Thus we see that the locally stratified programs do not permit any recursion through negation, whereas the Φ-accessible programs permit it to some extent, but control recursion through positive atoms in some ways. The weakly stratified programs and the programs with total well-founded model are easily seen to be extensions of the former program classes, in that they both allow recursion through positive atoms and restrict recursion through negative atoms. It is also easy to see from our new definitions with level mappings, that the well-founded semantics achieves this goal in a much more convenient way. The reason for this is, that the weakly stratified semantics can not handle recursion through positive atoms and recursion through negative atoms totally independently, as we have seen in the above example.

This explanation is very informal, its very purpose is just to give the reader an intuition of what the difference between the weakly stratified programs and the programs with total well-founded model is, and to show, how easy and convenient intuitive understanding of logic program classes can be obtained from level mapping based definitions and thus justify the alternative definitions given in Definitions 3.13 and 4.7.

We now see, that with the help of the alternative characterizations given in the thesis two more logic program classes can now be understood by means of level mappings. Thus large parts of the hierarchy of logic program classes can now be understood intuitively and are accessible to a variety of mathematical methods developed e.g. in [Hit01]. Nevertheless, there are some more things left to do. If we have a look at Definitions 3.13 and 4.7, there are some generalizations of these definitions one can think of.

The first generalization is concerned with Condition 1. As we have seen in Lemma 3.19, this condition assures the existence of a stable model. Thus a program class defined only by the first condition will have a stable model. It is to be investigated, whether the converse of this also holds, i.e. if all programs having a stable model, satisfy such a condition. Dropping or weakening the first condition in the Definitions 3.13 and 4.7 seems to destroy the program's property of having a stable model. If we e.g. admit a semi-strict decrease in the positive atoms, also the following program would be admitted:

5.4 Program

$$\begin{array}{rcl} A & \leftarrow & \neg A. \\ A & \leftarrow & A. \end{array}$$

This program clearly has no stable model, its unique supported model is $\{A\}$, but any level mapping $|\cdot|_4$ would satisfy the semi-strict decrease because of the second clause. Thus we see, that the weakening of Condition 1 is an insufficient approach to a level mapping based characterization of the programs with stable models. Another challenge in this direction is to investigate, whether the uniqueness of a stable model can also be expressed in such a semi-syntactic manner.

Other approaches for the application of the results obtained in this thesis are e.g. to generalize the semi-syntactic characterization for the well-founded semantics to the case of disjunctive logic programs. This would then yield a criterion to distinguish the several approaches to well-founded semantics proposed for disjunctive programs and to decide, which of them is the most natural generalization of the well-founded semantics for normal logic programs.

We want to note at last, that the results obtained so far, are only concerned with the two-valued case, i.e. the well-founded model resp. the weakly perfect model are supposed to be total in this thesis. A generalization of this to the three-valued case, i.e. characterizations of partial well-founded resp. weakly perfect models has been undertaken in [HW02].

References

[ABW88] K.R. Apt, H.A. Blair, and A. Walker. Towards a theory of declarative knowledge. In J. Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 89-148. Morgan Kaufmann, Los Altos, CA, 1988.

[AP93] K.R. Apt and D. Pedreschi. Reasoning about termination of pure prolog programs. *Information and Computation*, 106:109-157, 1993.

[BF91] N. Bidoit and C. Froidevaux. Negation by default and unstratifiable logic programs. *Theoretical Computer Science*, 75:85-112, 1991.

[Cla78] K.L. Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logic and Data Bases*, pages 293-322. Plenum Press, New York, 1978.

[Fag91] F. Fages. A new fixpoint semantics for general logic programs compared with the well-founded and the stable model semantics. *New Generation Computing*, 9:425-443, 1991.

[Fit85] M. Fitting. A Kripke-Kleene-semantics for general logic programs. *Journal of Logic Programming*, 2:295-312, 1985.

[GL88] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R.A. Kowalski and K.A. Bowen, editors, *Logic Programming. Proceedings of the 5th International Conference and Symposium on Logic Programming*, pages 1070-1080. MIT Press, 1988.

[GRS91] A. Van Gelder, K.A. Ross and J.S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620-650, 1991.

[Hei01] R. Heinze. Topological and Generalized Metric Methods for Logic Programming Semantics. Diploma thesis. Department of Computer Science, University of Bonn. 2001.

[Hit01] P. Hitzler. Generalized Metrics and Topology in Logic Programming Semantics. Ph.D. thesis. Department of Mathematics, National University of Ireland, University College Cork. 2001.

[HS99] P. Hitzler and A.K. Seda. Characterizations of classes of programs by three-valued operators. In M. Gelfond, N. Leone, and G. Pfeifer, editors, *Logic Programming and Nonmonotonic Reasoning, Proceedings of the 5th International Conference on Logic Programming and Non-monotonic Reasoning (LPNMR'99), El Paso, Texas, USA*, volume 1730 of *Lecture Notes in Artificial Intelligence*, pages 357-371. Springer, Berlin, 1999.

[HW02] P. Hitzler and M. Wendt. The Well-Founded Semantics is a Stratified Fitting Semantics. Submitted.

[Llo88] J.W. Lloyd. *Foundations of Logic Programming.* Springer, Berlin, 1988.

[Prz88] T.C. Przymusinski. On the declarative semantics of deductive databases and logic programs. In J. Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 193-216. Morgan Kaufmann, Los Altos, CA, 1988.

[PP90] H. Przymusinska and T.C. Przymusinski. Weakly stratified logic programs. *Fundamenta Informaticae*, 13:51-65, 1990. K.R. Apt, editor, special issue of Fundamenta Informaticae on Logical Foundations of Artificial Intelligence.