Topology and Iterates in Computational Logic*

Anthony Karel Seda & Pascal Hitzler

Department of Mathematics, University College, Cork, Ireland.
E-mail: aks@bureau.ucc.ie
Mathematische Fakultät, Universität Tübingen, Germany.
E-mail: pascal.hitzler@student.uni-tuebingen.de

Abstract

We consider the problem of finding models for logic programs P via fixed points of immediate consequence operators, T_P. Certain extensions of syntax invalidate the classical approach, adopted in the case of definite programs, using iterates of T_P and the Knaster-Tarski theorem. We discuss alternatives to the use of this theorem based on elementary notions from topological dynamics. This leads us to consider simple syntactic conditions on P, employing level mappings taking values in a countable ordinal γ, which ensure convergence (to models and fixed points) of the requisite sequences of iterates. We obtain, as a result, a constructive approach to the perfect model semantics of Przymusinski for locally stratified programs, somewhat along the lines of the approach adopted by Apt, Blair and Walker for stratified programs. In particular, when certain inequalities are sharp, we show the existence of unique supported models, which improves Przymusinski's results for perfect models. This result is obtained by viewing a Scott domain as a generalized ultrametric space, and applying a fixed-point theorem due to Priess-Crampe and Ribenboim. When γ happens to be ω, these results extend Fitting's treatment by metric methods of certain non-stratified programs discussed by Apt and Pedreschi in termination problems.

Subject Classifications: 68Q55, 54H25
Keywords: Logic programming semantics, level mapping, generalized ultrametric space, domain, topological dynamics.

1 Introduction

Computational logic is concerned with the use of logic as a programming language, and broadly consists of the following three components. (1) A syntax, or knowledge representation language, together with a theorem prover or interpreter. In this paradigm, program statements are viewed as axioms, and computation is viewed as deduction from the axioms via the theorem prover. (2) A distinguished minimal model M (a semantics) the purpose of which is to provide any program with its “intended meaning”.

*Topology Proceedings Vol. 22, Proceedings of the 12th Summer Conference on General Topology and its Applications: Special Session on Topology in Computer Science, Ontario, August 1997, pp. 1–43, to appear.

(3) An operator T with the property that M is a fixed point of T (perhaps the least fixed point or a minimal fixed point of T). Furthermore, one expects (1), (2) and (3) to be connected by a result expressing, on the one hand, completeness and soundness of the theorem prover and, on the other hand, expressing, in terms of T, some form of tractability in relation to the process of determining M.

The classic example of this is provided by *definite* or *positive* logic programs. In this case, the syntax is simply the Horn-clause subset of first order predicate logic together with SLD-resolution as the theorem prover. Thus, a definite program P consists of finitely many clauses of the form $A \leftarrow A_1, \ldots, A_n$ in which A and all the A_i are atoms, and $n \geq 0$; the case $n = 0$ is an abuse of notation indicating an empty antecedent or body i.e. a *unit clause* or *fact* $A \leftarrow$. Here, M is the least Herbrand model M_P, T is the immediate consequence operator T_P, and the requisite connection between the components is established by the following well-known theorem of Apt, Kowalski and van Emden, see [12], in which $lfp(T_P)$ denotes the least fixed point of T_P.

Theorem 1.1 *For any definite program P, we have $M_P = lfp(T_P) = T_P \uparrow \omega(\emptyset) = \{A \in B_P; P \models A\} = \{A \in B_P; P \vdash A\}$.* ■

It is worth drawing attention to the fact that the proof of this theorem depends on the lattice-continuity, and hence monotonicity, of T_P and on an application of the Knaster-Tarski theorem (the fixed-point theorem for complete partial orders).

Despite the rather restricted syntax, it turns out that any partial recursive (computable) function can be computed by some definite program P, so that the class of definite programs is computationally adequate. Nevertheless, there is a lot of current interest in the question of making definite programs more expressive and more flexible for programming purposes, and also in the question of modelling uncertain and non-monotonic reasoning etc. Such questions involve many technicalities, but in essence can be categorized under the following broad headings. (i) The extension of the syntax of definite programs. (ii) The enlargement of the set of truth values one uses to include, say, three, four, many or even infinitely-many truth values. (iii) Changing the underlying logic to permit non-classical logics.

In this paper, the extension of syntax we undertake is to include negated atoms in the bodies of clauses, so that we consider *normal* logic programs i.e. programs which consist of finitely many clauses of the form $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$. In such a clause, the symbols A, all the A_i and all the B_j are atoms, $k_1, l_1 \geq 0$ and the commas stand for conjunction i.e. $A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$ denotes $A_1 \wedge \cdots \wedge A_{k_1} \wedge \neg B_1 \wedge \cdots \wedge \neg B_{l_1}$. Moreover, the symbol "$\leftarrow$" denotes the logical connective of material implication. It is worth noting that this change to the syntax does indeed give a considerable gain in expressiveness, and this point is discussed in [1]. As far as issues (ii) and (iii) are concerned, we make no change and therefore we confine ourselves in this paper to just the two truth values *true* and *false*, and deal only with classical first order logic.

Even such a simple change as that we have just made to the syntax leads to the following problem.

Problem 1.2 (1) The operator T_P is no longer monotonic and therefore the Knaster-Tarski theorem is no longer applicable, yet it remains a central problem to find pre-fixed points and fixed points of T_P.
(2) Some form of Theorem 1.1 should still apply in the new context.

One way round the first of these problems is to define powers of the operator T_P in such a way as to recover monotonicity; this is the way adopted in [1] and it will be further discussed in this paper in §4. An alternative approach is to consider the extent to which the methods of Topology and Analysis can be used as a substitute for the Knaster-Tarski theorem. Indeed, work already undertaken in this direction includes the use of lattice topologies [3, 4]; the use of metrics and the Banach contraction mapping theorem [8], see also [7]; the use of metrics for multi-valued mappings in the case of disjunctive logic programs [11]; the use of the Rutten-Smyth fixed-point theorem for non-expansive operators on quasi-metric spaces [20].

The present paper is concerned with this alternative approach, and our main objective is to explore the use of elementary ideas from topological dynamics within the model theory of logic programs P. Thus, we concentrate on the issue (1) raised in Problem 1.2 and, for reasons which will become clear shortly, do not address (2) at all. In other words, we do not investigate the question of the existence of interpreters and their completeness and soundness in relation to model theory. Indeed, our specific aim is to use ideas connected with convergence of sequences of iterates to find models and supported models M for P. In fact, the former correspond to pre-fixed points of T_P (interpretations M satisfying $T_P(M) \subseteq M$), and the latter correspond to fixed points of T_P (M is supported if it satisfies $T_P(M) \supseteq M$, see [1]), and our thinking is based on the following simple observation.

Observation 1.3 Suppose P is a normal logic program and I is an interpretation for P. If the sequence of iterates $(T_P^n(I))_{n\in N}$ of I converges in the Cantor topology Q (see §2) to some M (it need not so converge), then M is a model for P but not necessarily a supported model. If, further, T_P is continuous in the Cantor topology (it need not be), then M is a supported model or fixed point of T_P.

Note 1.4 A similar fact holds for definite logic programs in relation to the Scott topology: Suppose P is a definite program and I an interpretation for P. Then the greatest limit M in the Scott topology of the sequence $(T_P^n(I))_{n\in N}$ of iterates of I is a model for P. If, further, the sequence of iterates is monotone increasing (it need not be if $I \neq \emptyset$), then M is a fixed point of T_P. The proof of this follows from [19, Theorem 6] and [20, Lemma 1] and employs the fact that T_P is always Scott continuous for any definite P. Indeed, taking I as $\emptyset$ permits one to recover the classical fixed-point theory for definite programs P, but this will not be discussed further here.

Observation 1.3 will be proved in §2, but for the moment we note that it raises the following question.

Question 1.5 (1) Can one provide conditions (necessary, sufficient or both necessary and sufficient) for the convergence of sequences $(T_P^n(I))_{n\in N}$ of iterates in the Cantor topology in terms of the syntax of P? In particular, can one do this when I is $\emptyset$?
(2) How general is Observation 1.3 as a means of finding pre-fixed points and fixed points of T_P?

In this paper, we propose to consider this question and to formulate answers to it in terms of level mappings l and inequalities between the values $l(A)$ and $l(A_i), l(B_j)$ in each ground instance $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$ of every clause in a normal logic program P, where l takes values in an arbitrary countable ordinal γ. Thus, in §2 we establish preliminaries and notation and formulate our main definition. Briefly, P is called (1) *level-decreasing*, respectively, (2) *strictly level-decreasing*, respectively, (3) *semi-strictly level-decreasing* if one has, respectively, the following inequalities holding for all i, j: (1) $l(A) \geq l(A_i), l(B_j)$, (2) $l(A) > l(A_i), l(B_j)$, (3) $l(A) \geq l(A_i), l(A) > l(B_j)$. In fact, see §2 below, the class of programs defined by (3) coincides exactly with the class of locally stratified programs defined by Przymusinski in [16] and in others of his many papers, see in particular [17, 18]. However, the terminology we adopt is more suited to our purposes since we intend to distinguish between the conditions (2) and (3) quite carefully, and the term "locally stratified" does not do this. As a matter of fact, the class of programs defined by Condition (1) is too general and will not be considered here in detail for the same reasons that it is not considered in [16], see §2. Przymusinski [17, 18] has discussed the existence of suitable interpreters for locally stratified programs and related them to model theory. For this reason, as already mentioned, we do not consider procedural semantics at all. Indeed, our results are entirely model-theoretic and may be summarized as follows. In §3 we examine the class defined by Condition (2). It turns out that in this case T_P is strictly contracting in the sense of Priess-Crampe and Ribenboim [14, 15] relative to a generalized ultrametric we define in terms of l, and which necessitates thinking of a Scott domain as a spherically complete generalized ultrametric space. We show, on using the fixed-point theorem of [14, 15], that in this case P has a unique supported model which coincides with the perfect model of [16]. This improves the results of Przymusinski to the extent that he showed uniqueness only of the perfect models. In particular, if l takes values in ω, then T_P is a contraction mapping relative to the ultrametric introduced by Fitting in [8]. We further explore this class in §3, briefly relating it to ideas of current interest in dynamical systems and computing being developed by Edalat in [6] and by us in more detail in [21]. Nevertheless, though of interest, the case $\gamma = \omega$ is too restrictive and it is essential to consider arbitrary countable ordinals γ for two reasons. First, doing this allows us to include arbitrary locally stratified programs within our framework. Second, one can show then that the class of strictly level-decreasing programs can compute all partial recursive functions, see [21], which is not the case if one is confined to ω-valued level mappings; some examples of programs which are strictly level-decreasing with respect to level mappings taking values in ordinals greater than ω are given in Example 3.11. Finally, in §4, we examine semi-strictly level-decreasing programs in depth. In this analysis we recover the perfect model semantics of [16]. However, what is new here is that our approach is very simple and constructive, see Construction 4.4, and our methods are rather different from those employed in [16]. Moreover, we establish recursion equations, see Corollary 4.6, which show very precisely how the iterates involved in the construction evolve. Finally, we note that another simplification obtained by this approach is that we work only with the ordinary iterates of T_P rather than with more complicated concepts such as the *powers* introduced in [1] and defined in §4 for convenience.

It is worth emphasizing the fact that the class of locally stratified programs forms a considerably larger class of programs than the stratified programs, containing, as it does, programs such as the "even numbers" program, see Example 3.12, and others considered by Fitting in [8] which are not stratified. Even the partial answer we give to Question 1.5 in this paper shows, therefore, that the ideas it embodies are a very general means indeed of finding models and supported models for logic programs.

Acknowledgement. The first named author acknowledges substantial support provided by the U.S. Office of Naval Research in presenting the results of this paper, and the second named author acknowledges the hospitality of University College Cork during its final preparation. We are grateful to an anonymous referee for suggesting several improvements in the style and presentation of the paper, and for drawing our attention to a flaw in our first version of Lemma 3.4.

2 Preliminaries and Notation

It will be convenient first to establish some preliminary concepts and definitions which will be used throughout the paper. Our notation is standard and follows [12]. Indeed, all undefined concepts relating to logic programming can be found in [12]. Thus, throughout the paper, P will denote an arbitrary normal logic program (as defined in the Introduction) whose underlying first order language will be denoted by L. We denote by B_P the *Herbrand base* of P i.e. the set of all ground or variable-free atoms in L. In fact, we shall usually suppose that L contains at least one function symbol of positive arity, so that B_P will usually be an infinite set. This assumption is not necessary, but without it topological considerations become rather trivial. Needless to say, all the results we establish apply in full generality whether or not L contains such a function symbol. Next, we let I_P denote the set of all Herbrand interpretations for P; as usual each Herbrand interpretation will be identified in a natural way with a subset of B_P, so that I_P is the power set $\mathcal{P}(B_P)$ of B_P. We use the notation ground(P) to denote the set of all ground instances of clauses in P i.e. the set of all instances $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$ of each clause in P in which A, A_i, B_j belong to B_P or, equivalently, contain no variable symbols, see [1]. As already noted, one of the most important concepts in the subject is that of the *immediate consequence operator* $T_P : I_P \to I_P$. This we define next by: $T_P(I) = \{A \in B_P;$ there is a clause $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1} \in$ ground(P) such that $I \models A_1 \wedge \cdots \wedge A_{k_1} \wedge \neg B_1 \wedge \cdots \wedge \neg B_{l_1}\}$. Notice that in classical two-valued logic, the statement $I \models A_1 \wedge \cdots \wedge A_{k_1} \wedge \neg B_1 \wedge \cdots \wedge \neg B_{l_1}$ is equivalent to the statement "$A_1, \ldots, A_{k_1} \in I$ and $B_1, \ldots, B_{l_1} \notin I$".

Finally, we let l denote a *level mapping* so that l is simply a mapping $l : B_P \to \gamma$, where γ denotes an arbitrary countable ordinal. In fact, γ will be regarded as the set of all ordinals n such that $n \in \gamma$ i.e. the set of ordinals n such that $n < \gamma$. As usual, if $n = m + 1$ is the successor of m, then we write $m = n - 1$ for the predecessor m of n. We call l an ω-level mapping in case $\gamma = \omega$, and also use the notation N for the set of natural numbers (including zero). We let $\mathcal{L}_n = \{A \in B_P; l(A) < n\}$, for $n \leq \gamma$, and put $\mathcal{L}_0 = \emptyset$. If $A \in B_P$ and $l(A) = n$, we say that *the level of A is n*. We call an ω-level mapping l *finite* if $\mathcal{L}_n$ is finite for each $n \in N$. Without loss of generality, we suppose always that the smallest value taken by l is zero.

Note that I_P can be naturally identified with $\mathbf{2}^{B_P}$, where $\mathbf{2}$ denotes the set $\{0, 1\}$. It can, therefore, be endowed with two well-known and important topologies. First, endow $\mathbf{2}$ with the Scott topology. Then, as is well-known, the product topology on I_P coincides with the Scott topology on I_P, viewed as a complete lattice, and it is this fact that underpins the observation made in Note 1.4. Second, endow $\mathbf{2}$ with the discrete topology. Then the product topology in this case makes I_P homeomorphic to the Cantor set. We shall denote this topology on I_P by Q and refer to it as the Cantor topology on I_P. Further details of these facts can be found in [19].

There is a simple criterion for convergence of sequences in Q. Again, this can be found in [19], but we include it here for completeness and in a slightly more comprehensive form, see [19, Proposition 4]. Typically, we denote sequences in I_P by $(I_n)_{n\in N}$ or by (I_n).

Proposition 2.1 *A sequence (I_n) in I_P is convergent iff for every $A \in B_P$ either A eventually belongs to I_n or A eventually does not belong to I_n (meaning that for all large enough n, $A \in I_n$ respectively $A \notin I_n$). If (I_n) is convergent, then its limit I is the set $\{A \in B_P; A$ eventually belongs to $I_n\}$.* ■

Using this proposition we can prove Observation 1.3.

Proof of Observation 1.3. Let I_n denote $T_P^n(I)$ and suppose that (I_n) converges in Q to M. For the first part, we must show that $T_P(M) \subseteq M$. Let $A \in T_P(M)$. Then by definition of T_P, there is a clause $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1} \in \text{ground}(P)$ such that, for all i, j, we have $A_i \in M$ and $B_j \notin M$. Since (I_n) converges to M in Q, there is, by Proposition 2.1 applied $(k_1 + l_1)$–times, an $n_0 \in N$ such that, for all $n \geq n_0$ and for all i, j, we have $A_i \in I_n$ and $B_j \notin I_n$. From this and the definition of T_P it follows that $A \in I_n$ for all $n \geq n_0 + 1$ and in turn it now follows from Proposition 2.1 again that $A \in M$.

Next, if T_P is continuous in Q, then a simple argument using the uniqueness of limits in Q, which is Hausdorff, shows that $T_P(M) = M$ as required.

Finally, taking P to be the following program:

$$r(o) \leftarrow$$

$$p(o) \leftarrow \neg r(o)$$
$$p(s(x)) \leftarrow p(x)$$
$$q(o) \leftarrow p(x)$$

and taking $I = \emptyset$ we find that (I_n) converges in Q to $M = \{r(o), q(o)\}$, yet $T_P(M) = \{r(o)\}$ so that M is not supported. ■

Of course, in the example just considered, T_P is not continuous in Q. Indeed, necessary and sufficient syntactic conditions for continuity of T_P were established in [19]. However, we will not make much use of continuity of T_P in this work, except in certain of the examples we discuss, and it will be enough to note that a sufficient condition for continuity is for P to contain no local variables, see [19, Corollary 6] (a variable (symbol) y is *local* if it occurs in the body of a clause but not in the head. For example, y is a local variable in the clause $p(x) \leftarrow p(y)$).

Level mappings as defined above have been used in a number of places in the literature on Logic Programming, where they have usually taken values in ω. For example, they have appeared in the study of termination problems, see [2, 5], in completeness problems, and in [8] to define metrics. We are now in a position to use them to formulate the main definition which we propose to consider in response to Question 1.5, and it will become apparent as we proceed that this paper builds on the work of [1, 8, 16, 17, 18].

Definition 2.2 Let P be a normal logic program, let $l : B_P \rightarrow \gamma$ be a level mapping and let $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$ denote a typical clause in ground(P). We call P:

(1) *Level-decreasing (with respect to l)* if the inequalities $l(A) \geq l(A_i), l(B_j)$ hold for all

i and j in each clause in ground(P).
(2) *Strictly level-decreasing (with respect to l)* if the inequalities $l(A) > l(A_i), l(B_j)$ hold for all i and j in each clause in ground(P).
(3) *Semi-strictly level-decreasing (with respect to l)* if the inequalities $l(A) \geq l(A_i)$ and $l(A) > l(B_j)$ hold for all i and j in each clause in ground(P).

As noted earlier in the Introduction, semi-strictly level-decreasing programs coincide exactly with the locally stratified programs defined in [16]. Indeed, if $l : B_P \to \gamma$ is a level mapping and we set $H_n = l^{-1}(n)$ for each ordinal $n < \gamma$, then in this way we set up a one-to-one correspondence between level mappings l and local stratifications $\{H_n; n < \gamma\}$. Of course, Class (2) is a strict subclass and Class (1) a strict superclass of the locally stratified programs. In fact, this latter class of programs, Class (1), can be disposed of immediately as being too general, and it will not be considered further. For example, it contains the program:

$$p(o) \leftarrow$$

$$p(s(o)) \leftarrow$$

$$p(x) \leftarrow \neg p(x)$$

and in this case T_P has no fixed points at all. Since T_P is continuous here, it follows from Observation 1.3 that the sequence $(T_P^n(I))$ can never converge in Q for any I. It was precisely in order to limit "recursion through negation" that stratified programs were introduced by Apt, Blair and Walker, see [1] and Van Gelder [24], and extended to locally stratified programs by Przymusinski in [16], and why the condition $l(A) > l(B_j)$ is imposed in (2) and (3) of Definition 2.2.

3 Strictly Level-Decreasing Logic Programs

The topology Q is of course metrizable, and indeed the following ultrametric d generates Q whenever we choose a finite level mapping $l : B_P \to \omega$ (see [20]): if $I_1 = I_2$, put $d(I_1, I_2) = 0$; otherwise, put $d(I_1, I_2) = 2^{-n}$, where I_1 and I_2 differ on some $A \in B_P$ such that $l(A) = n$, but agree on all atoms of lower level. This metric was introduced by Fitting in [8] where three problematic programs were discussed (the "even numbers" program, a "game" program and also a "transitive closures of graphs" program). In each case, it was shown that T_P is a contraction mapping and hence, by applying the Banach contraction mapping theorem, that each program has a unique supported model. Fitting also discussed a class of programs called "acceptable" by Apt and Pedreschi and encountered in discussions of termination problems in logic programming, see [2, 5]. Indeed, the definition of a strictly level-decreasing program relative to an ω-level mapping is implicit in Fitting's discussion of acceptable programs, although not explicitly given by him in [8]. Notice that the programs just mentioned are not stratified so that the methods of [1] are not applicable to them. It was precisely for this reason that Fitting introduced the metric d and applied the Banach contraction mapping theorem to discuss their semantics.

It turns out, see Theorem 3.9, that if P is strictly level-decreasing with respect to an ω-level mapping l, then T_P is a contraction mapping relative to the metric d determined by l, and hence that the Banach contraction mapping theorem may be applied to obtain a unique supported model for P. In fact, the main objective of this section of the paper, §3, is to establish a completely general version, Theorem 3.8, of Theorem 3.9. This result effectively disposes of the class of strictly level-decreasing programs and improves on the results of [16], as already noted in the Introduction, in that we show uniqueness of supported models and not just of perfect models (it emerges that perfect models are supported in fact). To obtain these results we make use of the fixed-point theorem of Priess-Crampe and Ribenboim [14, 15] in place of the Banach contraction mapping theorem (which is not directly applicable in this case). This necessitates showing that every domain can be turned into a generalized ultrametric space which is spherically complete in the sense of [14, 15], and the next few results are devoted to establishing the details of these facts.

Definition 3.1 (See [14, 15]) Let X be a set and let Γ be a partially ordered set with least element 0. We call the pair (X, d) a *generalized ultrametric space* if $d : X \times X \to \Gamma$ is a function satisfying the following conditions for all $x, y, z \in X$ and $\gamma \in \Gamma$.
(1) $d(x, y) = 0$ if and only if $x = y$.
(2) $d(x, y) = d(y, x)$.
(3) If $d(x, y) \leq \gamma$ and $d(y, z) \leq \gamma$, then $d(x, z) \leq \gamma$.

For $0 \neq \gamma \in \Gamma$ and $x \in X$, the set $B_\gamma(x) = \{y \in X; d(x, y) \leq \gamma\}$ is called a *γ-ball* or just a *ball* in X. A generalized ultrametric space is called *spherically complete* if, for any chain $\mathcal{C}$ of balls in X (i.e. for any set of balls which is totally ordered by inclusion), we have $\bigcap \mathcal{C} \neq \emptyset$. A function $f : X \to X$ is called *strictly contracting* if $d(f(x), f(y)) < d(x, y)$ for all $x, y \in X$ with $x \neq y$.

The following theorem was given in [14] and in more general form in [15].

Theorem 3.2 *Let (X, d) be a spherically complete generalized ultrametric space and let $f : X \to X$ be strictly contracting. Then f has a unique fixed point.*

In order to apply this theorem, we intend to show first that every Scott domain can be made into a spherically complete generalized ultrametric space.

Let $(D, \sqsubseteq)$ denote a Scott domain with set D_C of compact elements, see [23] for terms and notation. For a countable ordinal γ, let Γ_γ be the set $\{2^{-\alpha}; \alpha < \gamma\}$ of symbols $2^{-\alpha}$ with ordering $2^{-\alpha} < 2^{-\beta}$ if and only if $\beta < \alpha$.

Definition 3.3 Let $r : D_C \to \gamma$ be a function, called a *rank function*, and denote $2^{-\gamma}$ by 0. Define $d_r : D \times D \to \Gamma_{\gamma+1}$ by

$$d_r(x, y) = \inf\{2^{-\alpha}; c \sqsubseteq x \text{ if and only if } c \sqsubseteq y \text{ for every } c \in D_C \text{ with } r(c) < \alpha\}.$$

Then (D, d_r) is called the generalized ultrametric space *induced by r*.

Notice that the definition just made is closely related to [22, Example 5] which in turn was employed in [20].

It is straightforward to verify that (D, d_r) is indeed a generalized ultrametric space, and we proceed to show next that (D, d_r) is spherically complete. It will be necessary to impose one standing condition on the rank function r namely that, for each $x \in D$ and for each ordinal $\alpha < \gamma$, the set $\{c \in \mathrm{approx}(x); r(c) < \alpha\}$ is directed whenever it is non-empty. In what follows it will simplify matters to denote the ball $B_{2^{-\alpha}}(x)$ by $B_\alpha(x)$.

Lemma 3.4 *Let $B_\alpha(x) \subseteq B_\beta(y)$. Then the following statements hold.*
(1) $\{c \in \mathrm{approx}(x); r(c) < \beta\} = \{c \in \mathrm{approx}(y); r(c) < \beta\}$.
(2) $B_\alpha = \sup\{c \in \mathrm{approx}(x); r(c) < \alpha\}$ *and* $B_\beta = \sup\{c \in \mathrm{approx}(y); r(c) < \beta\}$ *both exist.*
(3) $B_\beta \sqsubseteq B_\alpha$.

Proof. Since $x \in B_\alpha(x)$, we have $x \in B_\beta(y)$ and hence $d_r(x, y) \leq 2^{-\beta}$. Therefore, the first statement follows immediately from the definition of d_r.

Since the set $\{c \in \mathrm{approx}(x); r(c) < \beta\}$ is bounded by x, for any x and β, the second statement follows from the consistent completeness of D.

For the third statement, suppose first that $B_\alpha(x) \subset B_\beta(y)$. Then we immediately have $\beta < \alpha$ by [14, (1.2)] since Γ_γ is totally ordered. Therefore, $B_\beta = \sup\{c \in \mathrm{approx}(y); r(c) < \beta\} = \sup\{c \in \mathrm{approx}(x); r(c) < \beta\} \sqsubseteq \sup\{c \in \mathrm{approx}(x); r(c) < \alpha\} = B_\alpha$, and so $B_\beta \sqsubseteq B_\alpha$ as required. Now suppose that $B_\alpha(x) = B_\beta(y) = B$, say. If $\alpha = \beta$, then it is immediate that $B_\alpha = B_\beta$. So suppose finally that $\alpha \neq \beta$ and suppose in fact that $\alpha < \beta$, so that $B_\alpha \sqsubseteq B_\beta$, with a similar argument if it is the case that $\beta < \alpha$. We intend to show again that $B_\alpha = B_\beta$, for which it suffices to obtain $d_r(B_\alpha, B_\beta) = 0$. By definition of d_r, B_α and B_β, we see that B_α and B_β are both elements of the ball B in question. Suppose that $d_r(B_\alpha, B_\beta) \neq 0$. Then there is a compact element c_1 such that the statement "$c_1 \sqsubseteq B_\alpha$ iff $c_1 \sqsubseteq B_\beta$" is false. Since $B_\alpha \sqsubseteq B_\beta$, it must be the case that $c_1 \not\sqsubseteq B_\alpha$ and $c_1 \sqsubseteq B_\beta$. By [14, (1.1)] any point of a ball is its centre, and so we can take y to be B_β in the equation established in Part (1). We therefore obtain $B_\beta = \sup\{c \in \mathrm{approx}(B_\beta); r(c) < \beta\}$. If $\{c \in \mathrm{approx}(B_\beta); r(c) < \beta\}$ is empty, then B_α and B_β are both equal to the bottom element of D and we are done; so suppose $\{c \in \mathrm{approx}(B_\beta); r(c) < \beta\} \neq \emptyset$. Since $c_1 \sqsubseteq B_\beta$, there is, by the condition imposed on r, a compact element c_2 with $r(c_2) < \beta$ such that $c_1 \sqsubseteq c_2 \sqsubseteq B_\beta$. But then $c_2 \not\sqsubseteq B_\alpha$ otherwise we would have $c_1 \sqsubseteq c_2$ and $c_2 \sqsubseteq B_\alpha$ leading to the contradiction $c_1 \sqsubseteq B_\alpha$. But now we have a compact element c_2 with $r(c_2) < \beta$ and for which $c_2 \not\sqsubseteq B_\alpha$ and $c_2 \sqsubseteq B_\beta$, and this contradicts the fact that $d_r(B_\alpha, B_\beta) \leq 2^{-\beta}$. Hence, $B_\alpha = B_\beta$ as required. ∎

Theorem 3.5 *Under the standing condition on r, (D, d_r) is spherically complete.*

Proof. By the previous lemma, every chain $(B_\alpha(x_\alpha))$ of balls in D gives rise to a chain (B_α) in D in reverse order. Let $B = \sup B_\alpha$. Now let $B_\alpha(x_\alpha)$ be an arbitrary ball in the chain. It suffices to show that $B \in B_\alpha(x_\alpha)$. Since $B_\alpha \in B_\alpha(x_\alpha)$, we have $d_r(B_\alpha, x_\alpha) \leq 2^{-\alpha}$. But d_r is a generalized ultrametric and so it suffices to show that $d_r(B, B_\alpha) \leq 2^{-\alpha}$. For every compact element $c \sqsubseteq B_\alpha$, we have $c \sqsubseteq B$ by construction of B. Now let $c \sqsubseteq B$ with $c \in D_C$ and $r(c) < \alpha$. We have to show that $c \sqsubseteq B_\alpha$. Since c is compact and $c \sqsubseteq B$, there exists B_β in the chain with $c \sqsubseteq B_\beta$. If $B_\alpha(x_\alpha) \subseteq B_\beta(x_\beta)$, then $B_\beta \sqsubseteq B_\alpha$ by Lemma 3.4 and therefore $c \sqsubseteq B_\alpha$. If $B_\beta(x_\beta) \subset B_\alpha(x_\alpha)$, then $\alpha < \beta$, and since $c \sqsubseteq B_\beta$, c is an element of the set $\{c \in \mathrm{approx}(x_\beta); r(c) < \alpha\} = \{c \in \mathrm{approx}(x_\alpha); r(c) < \alpha\}$. Since B_α is the supremum of the latter set, we have $c \sqsubseteq B_\alpha$ as required. ∎

To apply these results to logic programming, we regard I_P as a domain, under set inclusion, whose set of compact elements is the set I_C of all finite subsets of B_P, see [20] for related results. We note also that in the special case of the domain I_P, results similar to Theorem 3.5 were obtained in [15].

Definition 3.6 Let P be a normal logic program and let $l : B_P \to \gamma$ be a level mapping. We define the rank function r_l *induced* by l by setting $r_l(I) = \max\{l(A); A \in I\}$ for every $I \in I_C$, with I non-empty, and taking $r_l(\emptyset) = 0$. The generalized ultrametric obtained from a rank function in this way will be denoted by d_l.

Notice that the condition imposed on r is trivially satisfied by r_l, and the following proposition will make it easier to calculate distances which depend on r_l.

Proposition 3.7 *Let P be a normal logic program, let $l : B_P \to \gamma$ be a level mapping for P and let $I, J \in I_P$. Then $d_l(I, J) = \inf\{2^{-\alpha}; I \cap \mathcal{L}_\alpha = J \cap \mathcal{L}_\alpha\}$.*

Proof. Immediate by the observation that, for every $I \in I_P$, $I = \sup\{\{A\}; A \in I\}$. ■

Our main result in this section is the following theorem.

Theorem 3.8 *Let P be a normal logic program which is strictly level-decreasing with respect to a level mapping $l : B_P \to \gamma$. Then T_P is strictly contracting with respect to the generalized ultrametric d_l induced by l. Therefore, T_P has a unique fixed point and hence P has a unique supported model.*

Proof. Let $I_1, I_2 \in I_P$ and suppose that $d_l(I_1, I_2) = 2^{-\alpha}$.
Case 1. $\alpha = 0$.
Let $A \in T_P(I_1)$ with $l(A) = 0$. Since P is strictly level-decreasing, A must be the head of a unit clause in ground(P). From this it follows that $A \in T_P(I_2)$ also. By the same argument, if $A \in T_P(I_2)$ with $l(A) = 0$, then $A \in T_P(I_1)$. Therefore, $T_P(I_1) \cap \mathcal{L}_1 = T_P(I_2) \cap \mathcal{L}_1$, and hence we have

$$d_l(T_P(I_1), T_P(I_2)) \leq 2^{-1} < 2^{-0} = d_l(I_1, I_2)$$

as required.
Case 2. $\alpha > 0$.
In this case, I_1 and I_2 differ on some element of B_P with level α, but agree on all ground atoms of lower level. Let $A \in T_P(I_1)$ with $l(A) \leq \alpha$. Then there is a clause $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$ in ground(P), where $k_1, l_1 \geq 0$, such that for all k, j we have $A_k \in I_1$ and $B_j \notin I_1$. Since P is strictly level-decreasing and $I_1 \cap \mathcal{L}_\alpha = I_2 \cap \mathcal{L}_\alpha$, it follows that for all k, j we have $A_k \in I_2$ and $B_j \notin I_2$. Therefore, $A \in T_P(I_2)$. By the same argument, if $A \in T_P(I_2)$ with $l(A) \leq \alpha$, then $A \in T_P(I_1)$. Hence we have $T_P(I_1) \cap \mathcal{L}_{\alpha+1} = T_P(I_2) \cap \mathcal{L}_{\alpha+1}$, and it follows that

$$d_l(T_P(I_1), T_P(I_2)) \leq 2^{-(\alpha+1)} < 2^{-\alpha} = d_l(I_1, I_2)$$

as required.
Thus, T_P is strictly contracting. Therefore, by Theorem 3.2, T_P has a unique fixed point and therefore P has a unique supported model as claimed. ■

It is worth noting that the proof of Theorem 3.2, as given in [14, 15], is not constructive and does not provide the means of actually finding the fixed point. By contrast, the results of §4 and of Corollary 4.6, in particular, give constructions for the fixed point obtained by Theorem 3.8.

In the case that l is an ω-level mapping, the argument given in the proof of Theorem 3.8 can be given in exactly the same form with respect to the ultrametric d introduced by Fitting and defined earlier. In this case, the Banach contraction mapping theorem is sufficient to obtain the fixed point which results, and we have the following theorem.

Theorem 3.9 *Suppose P is strictly level-decreasing with respect to an ω-level mapping l. Then T_P is a contraction with respect to the ultrametric d with contractivity factor*[1] $\leq \frac{1}{2}$. *Therefore, T_P has a unique fixed point and hence P has a unique supported model.*

Staying with ω-level mappings for a moment, Fitting noted in [8] that when the Banach contraction mapping theorem applies, the fixed point it produces is obtained by considering iterates $T_P^n(I)$ for any $I \in I_P$, and that the sequence of iterates must close off by the first infinite ordinal (so that one does not need to enter the transfinite in this case). In particular, with $I = \emptyset$ we see that $\lim T_P^n(\emptyset)$ is a supported model. Moreover, Fitting noted that all the standard semantics for P (e.g. perfect model, stable model etc.) must coincide when T_P has a unique fixed point. Therefore, we have more generally the following corollary of Theorem 3.8 and Theorem 4.9.

Corollary 3.10 *Suppose that P is strictly level-decreasing with respect to an arbitrary level mapping $l : B_P \to \gamma$. Then all semantics for P coincide with the perfect model semantics of [16] which is the unique minimal supported model for P.* ■

Example 3.11 (1) Take P to be the following program:

$$q(o) \leftarrow \neg p(x), \neg p(s(x))$$

$$p(o) \leftarrow$$

$$p(s(x)) \leftarrow \neg p(x)$$

and define $l : B_P \to \omega + 1$ by $l(p(s^n(o))) = n$ and $l(q(s^n(o))) = \omega$ for all $n \in N$. Then P is strictly level-decreasing and the unique supported model given by Theorem 3.8 is the set $\{p(s^{2n}(o)); n \in N\}$.
(2) This time take P to be as follows:

$$p(o, o) \leftarrow$$

$$p(s(y), o) \leftarrow \neg p(y, x), \neg p(y, s(x))$$

$$p(y, s(x)) \leftarrow \neg p(y, x)$$

and define $l : B_P \to \omega\omega$ by $l(p(s^k(o), s^j(o))) = \omega k + j$, where ωk denotes the k^{th} limit ordinal. Then P is strictly level-decreasing and its unique supported model is the set $\{p(o, s^{2n}(o)); n \in N\} \cup \{p(s^{n+1}(o), s^{2k+1}(o)); k, n \in N\}$.

Example 3.12 Take P to be the "even numbers" program:

$$p(o) \leftarrow$$

$$p(s(x)) \leftarrow \neg p(x)$$

with the ω-level mapping l defined by $l(p(s^n(o))) = n$. Then Theorem 3.9 applies to this program (with contractivity factor $\frac{1}{2}$) and produces the set $\{p(o), p(s^2(o)), p(s^4(o)), \ldots\}$ of even numbers as the unique fixed point of T_P.

[1] This terminology is that of M. Barnsley, "Fractals Everywhere". Academic Press, Inc., San Diego, 1988.

Example 3.13 Consider the following program P:

$$p(s(o)) \leftarrow \neg q(o)$$

$$p(x) \leftarrow r(x)$$

$$r(x) \leftarrow p(x)$$

$$q(o) \leftarrow$$

The set $\{q(o), p(s^n(o)), r(s^n(o))\}$ is a fixed point of T_P for every n. Therefore, T_P can never satisfy the hypothesis of Theorem 3.8. In fact, this program is semi-strictly level-decreasing, but is never strictly level-decreasing for any level mapping because of the cycle created by the second and third clauses. Such a cycle would be prohibited in a strictly level-decreasing program, and this example shows that a semi-strictly level-decreasing program need not have a contractive immediate consequence operator.

Question 3.14 To what extent is the converse of Theorem 3.8 true? An answer to this question would set limits to the applicability of generalized ultrametrics determined by level mappings.

In fact, the strict converse of Theorem 3.8 is false, as shown by the following example.

Example 3.15 Take P as follows:

$$p(x) \leftarrow$$

$$p(x) \leftarrow p(s(x))$$

$$q(o) \leftarrow$$

$$q(s(x)) \leftarrow q(x)$$

In this case, T_P is a contraction with contractivity factor $\frac{1}{2}$ when we take l to be the ω-level mapping: $l(p(s^n(o))) = l(q(s^n(o))) = n$ for all $n \in N$. But because of the second clause, P is never strictly level-decreasing with respect to any level mapping. However, removing the second clause to obtain a program P' changes nothing i.e. $T_P = T_{P'}$ so that P and P' are *subsumption equivalent* as defined by Michael Maher in [13]. Thus, the previous question is modulo equivalences of this sort.

The results just discussed suggest connections between computational logic and dynamical systems, and we propose to briefly consider three of these next. This material is being included here in §3, but it is not assumed in what follows that P is necessarily strictly level-decreasing with respect to any level mapping.

First, let us fix a listing $B_P = \{A_0, A_1, A_2, \ldots\}$ of B_P and use it to determine the ω-level mapping l for the present; so that $l(A_n) = n$ for all n. Setting $\mathbf{2}_{A_i} = \mathbf{2}_i = \mathbf{2} = \{0, 1\}$ for all i, allows us to make the further identification of I_P with $\prod_{i=0}^{\infty} \mathbf{2}_i$ in which $I \in I_P$ corresponds to the sequence $\alpha = (\alpha_0, \alpha_1, \alpha_2, \ldots)$, where $\alpha_i = 1$ if $A_i \in I$ and equals 0 otherwise. Fitting's metric now coincides with one often used in symbolic dynamics: $d(\alpha, \beta) = 0$ if $\alpha = \beta$; otherwise $d(\alpha, \beta) = 2^{-n}$, where $n \geq 0$ is the smallest integer such that $\alpha_n \neq \beta_n$ and where of course $\alpha = (\alpha_0, \alpha_1, \alpha_2, \ldots)$ and $\beta = (\beta_0, \beta_1, \beta_2, \ldots)$ are elements of $\prod_{i=0}^{\infty} \mathbf{2}_i$. Furthermore, under this identification, T_P is conjugate to a sort of shift operator S_P on $\prod_{i=0}^{\infty} \mathbf{2}_i$.

Example 3.16 As an example of the foregoing comments, the program P_1:

$$p(x) \leftarrow p(s(x))$$

corresponds to the shift $(a_0, a_1, a_2, \ldots) \mapsto (a_1, a_2, a_3, \ldots)$ and therefore models chaotic behaviour to the same extent that this shift does this (notice that T_{P_1} and equivalently S_{P_1} has periodic points of every period). The program P_2:

$$p(s(x)) \leftarrow p(x)$$

corresponds to the shift $(a_0, a_1, a_2, \ldots) \mapsto (0, a_0, a_1, \ldots)$. The program P_3:

$$p(o) \leftarrow$$

$$p(s(x)) \leftarrow p(x)$$

corresponds to the shift $(a_0, a_1, a_2, \ldots) \mapsto (1, a_0, a_1, a_2, \ldots)$. And the program P_4 of Example 3.12:

$$p(o) \leftarrow$$

$$p(s(x)) \leftarrow \neg p(x)$$

corresponds to the mapping $(a_0, a_1, a_2, \ldots) \mapsto (1, 1 - a_0, 1 - a_1, 1 - a_2, \ldots)$.

For our second observation, we impose the mild condition that P contains at least one unit clause. It follows then that $T_P(\emptyset) \neq \emptyset$, and that we can choose the listing mentioned in the previous paragraph to satisfy the additional condition that $A_0 \in T_P(I)$ for every $I \in I_P$. Note that P is otherwise arbitrary and, in particular, we do not impose the condition on P that T_P be continuous in Q for what follows. Embed B_P into the unit interval $[0, 1]$ by defining $i(A_0) = 0$ and $i(A_n) = 2^{-n}$ for $n \geq 1$. Thus, B_P becomes a compact metric space. Let V_P denote the subspace of I_P consisting of all those elements of I_P which contain A_0, and endow V_P with the subspace topology of I_P. By virtue of Proposition 2.1, V_P is itself closed and hence compact, and moreover each element of V_P is a non-empty closed subset of B_P. In fact, the topology of V_P as a subspace of I_P coincides with that induced by the Hausdorff metric determined by the metric on B_P, so that V_P is a closed subspace of *Vietoris space*, see [6]. Finally, because $A_0 \in T_P(I)$ for all $I \in I_P$, we see that V_P is an invariant set under T_P. Thus, $T_P : V_P \to V_P$ is an abstract dynamical system, abstract in the sense that T_P need not be (usually is not) induced by a point map on B_P. Since $I_1 = T_P(I)$ belongs to V_P for any $I \in I_P$, iterates of I enter and stay within V_P. Thus, any model or fixed point which can be found by means of convergent sequences of iterates can be so found within V_P.

Example 3.17 The previous discussion raises the question of syntactic conditions under which T_P is a contraction relative to the Hausdorff metric. For example, the "natural numbers" program P as follows:

$$p(o) \leftarrow$$

$$p(s(x)) \leftarrow p(x)$$

has the property that T_P is such a contraction with the obvious listing of B_P. On the other hand, the "even numbers" program of Example 3.12 does not have this property.

For our third and final observation, suppose that $P = P_1 \cup \ldots \cup P_n$ is a partition of P into n sub-programs in which the definition of each predicate symbol is contained in one of the P_i (the *definition* of a predicate symbol p is the set of all clauses in P in which the predicate symbol p occurs in the head). We can then write T_P as the union $(\bigcup_{i=1}^n T_{P_i})$ in the sense that for all $I \in I_P$ we have $T_P(I) = (\bigcup_{i=1}^n T_{P_i})(I) = \bigcup_{i=1}^n T_{P_i}(I)$. In this representation, each of the T_{P_i} is to be thought of as a mapping of I_P into itself rather than as a mapping of I_{P_i} into itself.

By means of Proposition 2.1 we have the following result.

Proposition 3.18 *Suppose that P is partitioned as above. Then the following two statements hold.*
(1) T_P is continuous in Q at a point $I \in I_P$ iff each of the T_{P_i} is continuous in Q at I.
(2) Suppose that each of the T_{P_i} in the representation above is a contraction relative to Fitting's metric d with contractivity factor $c_i = 2^{-n_i}$, say. Then T_P is a contraction relative to d with contractivity factor $c = max\{c_i; i = 1, \ldots, n\}$. Conversely, if T_P is a contraction with factor of contractivity c relative to d, then each of the T_{P_i} is a contraction relative to d with contractivity factor $\leq c$. ■

Thus, whenever T_P is continuous in Q, $\{I_P; T_{P_1}, \ldots, T_{P_n}\}$ is an iterated function system which is in fact hyperbolic under the conditions of Proposition 3.18 (2).

Example 3.19 The program in Example 3.13 gives rise to an iterated function system which is never hyperbolic for any choice of level mapping l. In the program P:

$$q(o) \leftarrow$$

$$q(s^3(x)) \leftarrow p(x)$$

$$p(o) \leftarrow$$

$$p(s^2(x)) \leftarrow \neg p(x)$$

the definition of q has contractivity factor $\frac{1}{8}$, and the definition of p has contractivity factor $\frac{1}{4}$. Therefore, P determines a hyperbolic iterated function system with contractivity factor $\frac{1}{4}$.

Supposing, finally, that T_P is continuous in Q, let $F(I_P)$ denote the set of non-empty compact subsets of I_P endowed with the Hausdorff metric d_h induced by d, where d is the metric determined by a finite ω-level mapping l. Then, in the standard way, T_P induces a map $F_P : F(I_P) \to F(I_P)$ defined by $F_P(A) = \{T_P(I); I \in A\}$ which is a contraction with contractivity factor c if T_P is such on I_P. Thus, $F(I_P)$ is the *space of fractals over* I_P and F_P is induced from the iterated function system $\{I_P; T_{P_1}, \ldots, T_{P_n}\}$.

These three comments are suggestive of interesting connections between computational logic on the one hand and dynamical systems on the other. In fact, it is ongoing work of the authors to investigate certain notions of dynamical systems, such as attractors, from the point of view of computational logic, and vice-versa. In particular, these ideas are being developed with a view to relating this work to that of Edalat [6] in the context of uncertain (probabilistic) reasoning.

4 Semi-Strictly Level-Decreasing Logic Programs

In this section, we take up the study of the class of programs defined by (3) of the Definition 2.2 or, in other words, of the class of locally stratified programs, P. This study will be conducted, of course, from our current point of view of attempting to answer Question 1.5, and our main results, as already mentioned in the Introduction, concern a constructive approach to the perfect model semantics of [16].

We begin the details with an example showing that Condition (3) of Definition 2.2 is not, by itself, a necessary one for convergence in Q of sequences of iterates.

Example 4.1 Take the program P as follows:

$$p(x) \leftarrow p(x), \neg p(s(x))$$

$$p(o) \leftarrow$$

It is clear that P is never semi-strictly level-decreasing with respect to any level mapping l. However, the sequence of iterates $(T_P^n(\emptyset))$ becomes constant, after the first iterate, with value $\{p(o)\}$. Hence, this sequence trivially converges in Q to the value $\{p(o)\}$, which is a fixed point of T_P. Note, in fact, that T_P is continuous in Q in this case.

This example shows that (3), and therefore of course (2), in Definition 2.2 does not provide an entirely general answer to Question 1.5, not even when T_P is continuous in Q and not even for the case $I = \emptyset$. As a matter of fact, Example 3.13 shows that (3) does not provide a sufficient condition either for convergence in Q of sequences of iterates (not even when P is stratified, and Example 3.13 is stratified) since the iterates of $\emptyset$ in this case oscillate between the sets $\{q(o), p(s(o))\}$ and $\{q(o), r(s(o))\}$. Nevertheless, when levels are carefully controlled as in Construction 4.4 below, (3) does provide a sufficient condition for convergence and this fact is used at an important point in the proof of Lemma 4.5 below.

Our approach is closer in spirit to [1] than it is to [16]. In fact, we will make comparisons on several occasions between our results and those of [1]. It therefore will be convenient for the reader if we recall next the notion of stratification as defined in [1] and to record the basic facts and notation used in the construction of the model M_P discussed there.

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

(1) If a predicate symbol occurs positively in a clause in P_i, then its definition is contained within $\bigcup_{j \leq i} P_j$.

(2) If a predicate symbol occurs negatively in a clause in P_i, then its definition is contained within $\bigcup_{j < i} P_j$.

We adopt the convention that the definition of a predicate symbol p occurring in P is contained in P_1 whenever its definition is empty. Thus, each predicate symbol occurring in P is defined but it may have empty definition; in particular, P_1 itself may be empty.

In order to treat non-monotonic operators, the *powers* of an operator T mapping a complete lattice into itself were defined as follows:

$T\uparrow 0(I) = I$
$T\uparrow(n+1)(I) = T(T\uparrow n(I)) \cup T\uparrow n(I)$
$T\uparrow\omega(I) = \bigcup_{n=0}^{\infty} T\uparrow n(I)$.

Of course, $T\uparrow n(I)$ is *not* equal to $T^n(I)$ unless T is monotonic. Indeed, the sequence $(T\uparrow n(I))$ is always monotonic increasing. However, this concept can be used to construct a minimal supported model M_P for any stratified program P as follows: put $M_0 = \emptyset, M_1 = T_{P_1}\uparrow\omega(M_0), \ldots, M_m = T_{P_m}\uparrow\omega(M_{m-1})$. Finally, let $M_P = M_m$. ■

4.1 The Case of Arbitrary Level Mappings

We commence with the following simple proposition which in fact is [16, Proposition 5]. However, we include a proof since we need certain details later.

Proposition 4.2 *Every stratified logic program is semi-strictly level-decreasing.*

Proof. Let $P = P_1 \cup \ldots \cup P_m$ be a stratification of P. We define an ω-level mapping l by $l(A) = i$ if A is a ground atom whose predicate symbol p, say, in L is defined in P_{i+1}. It is clear that P is semi-strictly level-decreasing with respect to l. ■

Notice that the level mapping defined in the proof just given is not, in general, finite and we will take up this issue later on.

Definition 4.3 Let P denote a normal logic program and let $l : B_P \to \gamma$ denote a level mapping, where $\gamma > 1$. For each n satisfying $0 < n < \gamma$, let $P_{[n]}$ denote the set of all clauses in ground(P) in which only atoms A with $l(A) < n$ occur. We define $T_{[n]} : \mathcal{P}(\mathcal{L}_n) \to \mathcal{P}(\mathcal{L}_n)$ by $T_{[n]}(I) = T_{P_{[n]}}(I)$. The mapping $T_{[n]}$ is called the *immediate consequence operator restricted at level n.*

Thus, the idea formalized by this definition is to "cut-off" at level n.

Construction 4.4 Let P be a semi-strictly level-decreasing normal logic program and let $l : B_P \to \gamma$ denote a level mapping, where $\gamma > 1$. We construct the transfinite sequence $(I_n)_{n\in\gamma}$ inductively as follows. For each $m \in N$, we put $I_{[1,m]} = T_{[1]}^m(\emptyset)$ and set $I_1 = \bigcup_{m=0}^{\infty} I_{[1,m]}$. If $n \in \gamma$, where $n > 1$, is a successor ordinal, then for each $m \in N$ we put $I_{[n,m]} = T_{[n]}^m(I_{n-1})$ and set $I_n = \bigcup_{m=0}^{\infty} I_{[n,m]}$. If $n \in \gamma$ is a limit ordinal, we put $I_n = \bigcup_{m<n} I_m$. Finally, we put $I_{[P]} = \bigcup_{n<\gamma} I_n$.

The main technical lemma we need is as follows. For its proof, which is by transfinite induction, it will be convenient to put $I_{[n,m]} = I_n$ for all $m \in N$ whenever n is a limit ordinal; thus statement (b) in the lemma makes sense for all ordinals n.

Lemma 4.5 *Let P be a normal logic program which is semi-strictly level-decreasing with respect to the level mapping $l : B_P \to \gamma$, where $\gamma > 1$. Then the following statements hold.*
(a) The sequence $(I_n)_{n\in\gamma}$ is monotonic increasing in n.
(b) For every $n \in \gamma$, where $n \geq 1$, the sequence $(I_{[n,m]})$ is monotonic increasing in m.
(c) For every $n \in \gamma$, where $n \geq 1$, I_n is a fixed point of $T_{[n]}$.
(d) If $l(B) < n$ and $B \notin I_n$, where $B \in B_P$, then for every $m \in \gamma$ with $n < m$ we have $B \notin I_m$ and hence $B \notin I_{[P]}$. In particular, if $l(B) < n$ and $B \notin I_{[n+1,m]}$ for some

$m \in N$, then $B \notin I_n$ and hence $B \notin I_{[P]}$.

Proof. It is immediate from the construction that the sequence $(I_n)_{n\in\gamma}$ is monotonic increasing in n, and this establishes (a).

The main work is in establishing (b) and (c), which we treat simultaneously. To do this, we need to note the technical fact that, for each $n \in \gamma$, we can partition $P_{[n+1]}$ as $P_{[n]} \cup P(n)$, where $P(n)$ denotes the subset of ground(P) consisting of those clauses whose head has level n. Thus, $T_{[n+1]}(I) = T_{[n]}(I) \cup T_{P(n)}(I)$ for any $I \in I_P$; note that if $A \in T_{P(n)}(I)$, then $l(A) = n$.

Let $\mathcal{P}(n)$ be the proposition, depending on the ordinal n, that $(I_{[n,m]})$ is monotonic increasing in m and that I_n is a fixed point of $T_{[n]}$. Suppose that $\mathcal{P}(n)$ holds for all $n < \alpha$, where $\alpha \in \gamma$ is some ordinal. We must show that $\mathcal{P}(\alpha)$ holds. Indeed, $\mathcal{P}(1)$ holds since $P_{[1]}$ is a definite program and the construction of I_1 is simply the classical construction of the least fixed point of $T_{[1]}$, and therefore we may assume that $\alpha > 2$. It will be convenient to break up the details of the case when α is a successor ordinal into a sequence of steps.

Case 1. $\alpha = k + 1$ is a successor ordinal. Thus, $\mathcal{P}(k)$ holds.

Step 1. We establish the recursion equations:

$$I_{[k+1,0]} = I_k$$

$$I_{[k+1,m+1]} = I_k \cup T_{P(k)}(I_{[k+1,m]})$$

and the first is immediate. Putting $m = 0$, we have $I_{[k+1,1]} = T_{[k+1]}(I_k) = T_{[k]}(I_k) \cup T_{P(k)}(I_k) = I_k \cup T_{P(k)}(I_k) = I_k \cup T_{P(k)}(I_{[k+1,0]})$, using the fact that I_k is a fixed point of $T_{[k]}$. Now suppose that the second of these equations holds for some $m > 0$. Then $I_{[k+1,(m+1)+1]} = T_{[k+1]}(I_{[k+1,m+1]}) = T_{[k]}(I_{[k+1,m+1]}) \cup T_{P(k)}(I_{[k+1,m+1]}) = T_{[k]}(I_k \cup T_{P(k)}(I_{[k+1,m]})) \cup T_{P(k)}(I_{[k+1,m+1]})$, and it suffices to show that $T_{[k]}(I_k \cup T_{P(k)}(I_{[k+1,m]})) = I_k$. So suppose that $A \in T_{[k]}(I_k \cup T_{P(k)}(I_{[k+1,m]}))$. Thus, there is a clause in $P_{[k]}$ of the form $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$ where $A_1, \ldots, A_{k_1} \in I_k \cup T_{P(k)}(I_{[k+1,m]})$ and $B_1, \ldots, B_{l_1} \notin I_k \cup T_{P(k)}(I_{[k+1,m]})$. But then level considerations and the hypothesis concerning P imply that $A_1, \ldots, A_{k_1} \in I_k$ and $B_1, \ldots, B_{l_1} \notin I_k$. Therefore, $A \in T_{[k]}(I_k) = I_k$ and we have the inclusion $T_{[k]}(I_k \cup T_{P(k)}(I_{[k+1,m]})) \subseteq I_k$. The reverse inclusion is demonstrated in like fashion, showing that the second of the recursion equations holds with m replaced by $m + 1$ and hence, by induction on m, that it holds for all m.

Step 2. We have the inclusions $T_{P(k)}(I_k) \subseteq T_{P(k)}(I_k \cup T_{P(k)}(I_k)) \subseteq T_{P(k)}(I_k \cup T_{P(k)}(I_k \cup T_{P(k)}(I_k))) \ldots$

These inclusions are established by methods similar to those we have just employed and we omit the details.

It is now clear from this fact and the recursion equations in Step 1 that $(I_{[k+1,m]})$, or $(I_{[\alpha,m]})$, is monotonic increasing in m. Since monotonic increasing sequences converge to their union in Q, see [19, Proposition 9], and $I_{[k+1,m]}$ is an iterate of I_k, it now follows from Observation 1.3 that I_{k+1} is a model for $P_{[k+1]}$.

Step 3. If $B \in B_P$ and $l(B) < k$, then $B \in I_{k+1}$ iff $B \in I_k$.

Indeed, if $B \in I_k$, then it is clear from the recursion equations of Step 1 that $B \in I_{k+1}$. On the other hand, if $B \notin I_k$, then it is equally clear from the recursion equations and level considerations that, for every $m \in N$, $B \notin I_{[k+1,m]}$ and hence that $B \notin I_{k+1}$, as required.

Step 4. I_{k+1} is a supported model for $P_{[k+1]}$.
To see this, suppose that $A \in I_{k+1} = \bigcup_{m=0}^{\infty} I_{[k+1,m]}$. Then there is $m_0 \in N$ such that $A \in I_{[k+1,m+1]} = T_{[k+1]}^{m+1}(I_k)$ for all $m \geq m_0$. Thus, $A \in T_{[k+1]}(T_{[k+1]}^{m_0}(I_k)) = T_{[k+1]}(I_{[k+1,m_0]})$. Hence, there is a clause $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$ in $P_{[k+1]}$ such that each $A_i \in I_{[k+1,m_0]}$ and no $B_j \in I_{[k+1,m_0]}$. But $l(B_j) < k$ for each j since P is semi-strictly level-decreasing. Since $B_j \notin I_{[k+1,m_0]}$, we now see from the recursion equations that $B_j \notin I_k$. From the result in Step 3 we now deduce that, for each j, $B_j \notin I_{k+1}$. Since it is obvious that each A_i belongs to I_{k+1}, we obtain that $A \in T_{[k+1]}(I_{k+1})$. Thus, $I_{k+1} \subseteq T_{[k+1]}(I_{k+1})$ and therefore I_{k+1} is a supported model for $P_{[k+1]}$, or a fixed point of $T_{[k+1]}$, as required.

Thus, $\mathcal{P}(\alpha)$ holds when α is a successor ordinal.

Case 2. α is a limit ordinal.
In this case, it is trivial that $(I_{[\alpha,m]})$ is monotonic increasing in m. Thus, we have only to show that I_α is a fixed point of $T_{[\alpha]}$ i.e. a supported model for $P_{[\alpha]}$, and we show first that I_α is a model for $P_{[\alpha]}$. Let $A \in T_{[\alpha]}(I_\alpha)$. Then there is a clause $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$ in $P_{[\alpha]}$ such that $A_1, \ldots, A_{k_1} \in I_\alpha$ and $B_1, \ldots, B_{l_1} \notin I_\alpha$. Indeed, by the definition of $P_{[\alpha]}$ and the hypothesis concerning P, there is $n_0 < \alpha$ such that the clause $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$ belongs to $P_{[n_0]}$. Since the sequence $(I_n)_{n\in\gamma}$ is monotone increasing and $I_\alpha = \bigcup_{n<\alpha} I_n$, there is $n_1 < \alpha$ such that $A_1, \ldots, A_{k_1} \in I_{n_1}$ and $B_1, \ldots, B_{l_1} \notin I_{n_1}$. Choosing $n_2 = \max\{n_0, n_1\}$, we have $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1} \in P_{[n_2]}$ and also $A_1, \ldots, A_{k_1} \in I_{n_2}$ and $B_1, \ldots, B_{l_1} \notin I_{n_2}$. Therefore, on using the induction hypothesis we have $A \in T_{[n_2]}(I_{n_2}) = I_{n_2} \subseteq I_\alpha$. Hence, $T_{[\alpha]}(I_\alpha) \subseteq I_\alpha$, as required.

To see that I_α is supported, let $A \in I_\alpha$. By monotonicity of $(I_n)_{n\in\gamma}$ again and the identity $I_\alpha = \bigcup_{n<\alpha} I_n$, there is a successor ordinal $n_0 \geq 1$ such that $A \in I_n$ for all n such that $n_0 \leq n < \alpha$. In particular, we have $A \in I_{n_0} = \bigcup_{m=0}^{\infty} I_{[n_0,m]}$. Therefore, there is $m_1 \in N$ such that $A \in I_{[n_0,m_1+1]} = T_{[n_0]}(T_{[n_0]}^{m_1}(I_{n_0-1}))$. Consequently, there is a clause $A \leftarrow A_1, \ldots, A_{k_1}, \neg B_1, \ldots, \neg B_{l_1}$ in $P_{[n_0]}$ such that $A_1, \ldots, A_{k_1} \in T_{[n_0]}^{m_1}(I_{n_0-1}) = I_{[n_0,m_1]} \subseteq I_{n_0} \subseteq I_\alpha$ and $B_1, \ldots, B_{k_1} \notin I_{[n_0,m_1]}$. But $l(B_j) < n_0 - 1$ for each j and so no B_j belongs to I_{n_0-1} by Step 3 of the previous case. Therefore, by this step, no B_j belongs to I_{n_0} and by iterating this we see that, for every $m \in N$, no B_j belongs to I_{n_0+m}. Therefore, no B_j belongs to I_α. Hence, we have $A \in T_{[n_0]}(I_\alpha) \subseteq T_{[\alpha]}(I_\alpha)$ or in other words that $I_\alpha \subseteq T_{[\alpha]}(I_\alpha)$, as required.

It now follows that $\mathcal{P}(n)$ holds for all ordinals n, and this completes the proof of (b) and (c). In particular, we see that the recursion equations obtained in Step 1 hold for all ordinals k, and we record this fact in the corollary below. Indeed, all that is needed to establish these equations is the fact that each I_k is a fixed point of $T_{[k]}$, and to note that the proof just given shows also that $I_{[P]}$ is a fixed point of T_P. In turn, (d) of the lemma now follows from this observation by iterating Step 3.

The proof of the lemma is therefore complete. ■

It can be seen here, and it will be seen again later, that the importance of (d) is the control it gives over negation in the manner illustrated in the proof just given that I_{k+1} is a supported model for $P_{[k+1]}$. It is also worth noting that the construction produces a monotonic increasing sequence by means of a non-monotonic operator, and that Lemma 4.5 plays a rôle here similar to that played by [1, Lemma 10] in [1].

Corollary 4.6 *Suppose the hypotheses of Lemma 4.5 all hold. Then:*
(1) For all ordinals n and all $m \in N$ we have the recursion equations

$$I_{[n+1,0]} = I_n$$

$$I_{[n+1,m+1]} = I_n \cup T_{P(n)}(I_{[n+1,m]}).$$

(2) If P is in fact strictly level-decreasing, then for every ordinal $n \geq 1$ we have $I_{[n+1,m]} = I_n \cup T_{P(n)}(I_n)$ for all $m \in N$, and therefore the iterates stabilize after one step.

Proof. That (1) holds has already been noted in the proof of Lemma 4.5. For (2), it suffices to prove that $T_{P(n)}(I_n) = T_{P(n)}(I_n \cup T_{P(n)}(I_n))$. So suppose therefore that $A \in T_{P(n)}(I_n \cup T_{P(n)}(I_n))$. Then there is a clause $A \leftarrow A_1, \dots, A_{k_1}, \neg B_1, \dots, \neg B_{l_1}$ in $P(n)$ such that $A_1, \dots, A_{k_1} \in I_n \cup T_{P(n)}(I_n)$ and $B_1, \dots, B_{k_1} \notin I_n \cup T_{P(n)}(I_n)$. From these statements and by level considerations, we have $A_1, \dots, A_{k_1} \in I_n$ and $B_1, \dots, B_{k_1} \notin I_n$. Therefore, $A \in T_{P(n)}(I_n)$ so that $T_{P(n)}(I_n \cup T_{P(n)}(I_n)) \subseteq T_{P(n)}(I_n)$. The reverse inclusion is established similarly to complete the proof. ■

Statement (2) of this corollary makes the calculation of iterates very easy to perform in the case of strictly level-decreasing programs, and an illustration of this is to be found in Example 4.13.

Theorem 4.7 *Suppose that P is a normal logic program which is semi-strictly level-decreasing with respect to the level mapping $l : B_P \to \gamma$. Then $I_{[P]}$ is a minimal supported model for P.*

Proof. That $I_{[P]}$ is a supported model for P follows from the proof of Lemma 4.5, and so it remains to show that $I_{[P]}$ is minimal. To do this, we establish by transfinite induction the following proposition: "if $J \subseteq I_{[P]}$ and $T_P(J) \subseteq J$, then $I_n \subseteq J$ for all $n \in \gamma$, where $n \geq 1$", and this clearly suffices. Indeed, $T_{[1]}(J) \subseteq T_P(J) \subseteq J$ and therefore J is a model for $P_{[1]}$. But, as already noted in proving Lemma 4.5, I_1 is the least model for $P_{[1]}$ by construction, since $P_{[1]}$ is definite. Therefore, $I_1 \subseteq J$ and the proposition holds with $n = 1$.

Now assume that the proposition holds for all ordinals $n < \alpha$ for some ordinal $\alpha \in \gamma$, where $\alpha > 1$; we show that it holds with $n = \alpha$.
Case 1. $\alpha = k + 1$ is a successor ordinal, where $k > 0$.
We have $I_k \subseteq J$. We show by induction on m that $I_{[k+1,m]} \subseteq J$ for all m. Indeed, with $m = 0$ we have $I_{[k+1,0]} = I_k \subseteq J$. Suppose, therefore, that $I_{[k+1,m_0]} \subseteq J$ for some $m_0 > 0$. Let $A \in I_{[k+1,m_0+1]} = T_{[k+1]}(T^{m_0}_{[k+1]}(I_k))$. Then there is a clause $A \leftarrow A_1, \dots, A_{k_1}, \neg B_1, \dots, \neg B_{l_1}$ in $P_{[k+1]}$ such that $A_1, \dots, A_{k_1} \in T^{m_0}_{[k+1]}(I_k) = I_{[k+1,m_0]}$ and $B_1, \dots, B_{l_1} \notin I_{[k+1,m_0]}$. But $l(B_j) < k$ for each j. Applying Lemma 4.5 (d) we see that no B_j belongs to $I_{[P]}$ and consequently no B_j belongs to J because $J \subseteq I_{[P]}$. Since $I_{[k+1,m_0]} \subseteq J$ by assumption, we have $A_1, \dots, A_{k_1} \in J$. Therefore, $A \in T_{[k+1]}(J) \subseteq T_P(J) \subseteq J$, and from this we obtain that $I_{[k+1,m_0+1]} \subseteq J$ as required to complete the proof in this case.
Case 2. α is a limit ordinal.
In this case, $I_\alpha = \bigcup_{n<\alpha} I_n$ and $I_n \subseteq J$ for all $n < \alpha$ by hypothesis. Therefore, $I_\alpha \subseteq J$ as required.

Thus, the result follows by transfinite induction. ■

The following definition is due to Przymusinski and is to be found in [16].

Definition 4.8 Suppose that P is a locally stratified normal logic program, and let l denote the associated level mapping. Given two distinct models M and N for P, we say that N is *preferable* to M if, for every ground atom A in $N \setminus M$, there is a ground atom B in $M \setminus N$ such that $l(A) > l(B)$. Finally, we say that a model M for P is *perfect* if there are no models for P preferable to M.

Notice that the requirement $l(A) > l(B)$ is dual to the requirement $A < B$ relative to the priority relation $<$ defined in [16].

Theorem 4.9 *Suppose that P is a normal logic program which is semi-strictly level-decreasing with respect to a level mapping $l : B_P \to \gamma$, where γ is a countable ordinal. Then $I_{[P]}$ is a perfect model for P and indeed is the only perfect model for P.*

Proof. Suppose that there is a model N for P which is preferable to $I_{[P]}$ (and therefore distinct from $I_{[P]}$); we will derive a contradiction.

First note that $N \setminus I_{[P]}$ must be non-empty, otherwise we have $N \subseteq I_{[P]}$. But this inclusion forces equality of N and $I_{[P]}$ since $I_{[P]}$ is a minimal model for P, and therefore N and $I_{[P]}$ are not distinct. This means that there is a ground atom A in $N \setminus I_{[P]}$, which can be chosen so that $l(A)$ has minimum value; let B be a ground atom in $I_{[P]} \setminus N$ corresponding to A in accordance with the definition above, and which satisfies $l(A) > l(B)$.

Next we note that $T_{[1]}(N) \subseteq T_P(N) \subseteq N$, since N is a model for P. Hence, N is a model for $P_{[1]}$, which implies that $I_1 \subseteq N$ since I_1 is the least model for the definite program $P_{[1]}$. Therefore, B can be chosen so that $B \in I_{n_0} \setminus N$, with minimal $n_0 > 1$. Now n_0 cannot be a limit ordinal, otherwise we would have $I_{n_0} = \bigcup_{m<n_0} I_m$, from which we would conclude that $B \in I_m \setminus N$ for some $m < n_0$ contrary to the choice of n_0. Thus, n_0 must be a successor ordinal and, therefore, B can be chosen so that $B \in I_{[n_0,m_0]} \setminus N$, where m_0 is such that $I_{[n_0,m_1]} \setminus N = \emptyset$ whenever $m_1 < m_0$; indeed, since $I_1 \subseteq N$, we must have $n_0 > 1$ and $m_0 \geq 1$ also. Consequently, $B \in T_{[n_0]}(I_{[n_0,m_0-1]}) \setminus N$ showing that there is a clause $B \leftarrow C_1, \ldots, C_{k_1}, \neg D_1, \ldots, \neg D_{l_1}$ in $P_{[n_0]}$ with the property that each $C_i \in I_{[n_0,m_0-1]}$ and no $D_j \in I_{[n_0,m_0-1]}$. Since $l(D_j) < n_0 - 1$ for each j, we see that none of the D_j belong to $I_{[P]}$ by Lemma 4.5 (d). But all the C_i, if there are any, must belong to N by the choice of the numbers n_0 and m_0. Moreover, there must be at least one D_j and indeed at least one belonging to N. For if there were no D_j or we had each $D_j \notin N$, then we would have $B \in T_{P_{n_0}}(N) \subseteq T_P(N) \subseteq N$, using again the fact that N is a model for P. But this leads to the conclusion that $B \in N$, which is contrary to $B \in I_{[P]} \setminus N$. Thus, there is a $D = D_j \in N \setminus I_{[P]}$, for some j, satisfying $l(D) < l(B) < l(A)$. Since A was chosen in $N \setminus I_{[P]}$ to have smallest level, we have a contradiction.

This contradiction shows that $I_{[P]}$ must be a perfect model for P as required. The last statement in the theorem concerning uniqueness of $I_{[P]}$ now follows from [16, Theorem 4], and therefore the proof is complete. ■

Since it is shown in [16] that perfect models are independent of the local stratification, we also have the following result.

Corollary 4.10 *If P is a normal logic program which is semi-strictly level-decreasing with respect to two level mappings l_1 and l_2, then the corresponding models $I_{[P_1]}$ and $I_{[P_2]}$ are equal.*

It also follows from [16, Theorem 4] and Theorem 4.9 above that $I_{[P]}$ coincides with the model M_P of [1] when P is stratified. However, for the sake of completeness we next present a proof of this fact using the methods established thus far. To do this, it will be convenient to introduce the concept $T \Uparrow n(I)$ for a mapping $T : I_P \to I_P$ and $I \in I_P$. In fact, $T \Uparrow n(I)$ is defined inductively as follows:
$T \Uparrow 0(I) = I$
$T \Uparrow (n+1)(I) = T(T \Uparrow n(I)) \cup I$
$T \Uparrow \omega(I) = \bigcup_{n=0}^{\infty} T \Uparrow n(I)$.

Theorem 4.11 *Let P be a stratified normal logic program with level mapping defined as in the proof of Proposition 4.2. Then $I_{[P]} = M_P$.*

Proof. As usual, we take the stratification to be $P = P_1 \cup \ldots \cup P_m$ and we will show by induction that $I_k = M_k$ for $k = 1, \ldots, m$ and that $I_k = M_m$ for $k > m$. From this we clearly have $I_{[P]} = M_m = M_P$ as required.

With the definition of the level mapping we are currently using and with the conventions we have made regarding the stratification, we note first that the equalities $P_{[k]} = \text{ground}(P_1 \cup P_2 \cup \ldots \cup P_k)$ and $P(k-1) = \text{ground}(P_k)$ both hold for $k = 1, \ldots, m$, where $P(k)$ is as defined in the proof of Lemma 4.5.

Now $P_{[1]} = \text{ground}(P_1)$ is definite, even if empty, and so it is immediate that $T_{P_1} \Uparrow i(M_0) = T_{P_1} \uparrow i(M_0)$ for all i and that $I_1 = M_1$. So suppose next that $T_{P_{k+1}} \Uparrow i(M_k) = T_{P_{k+1}} \uparrow i(M_k)$ for all i and that $I_{k+1} = M_{k+1}$ for some $k > 0$. Then $T_{P_{k+2}} \Uparrow 0(M_{k+1}) = M_{k+1} = T_{P_{k+2}} \uparrow 0(M_{k+1})$ and also $I_{[k+2,0]} = I_{k+1} = M_{k+1} = T_{P_{k+2}} \uparrow 0(M_{k+1})$. So now suppose that $T_{P_{k+2}} \Uparrow m(M_{k+1}) = T_{P_{k+2}} \uparrow m(M_{k+1})$ and that $I_{[k+2,m]} = T_{P_{k+2}} \uparrow m(M_{k+1})$ for some $m > 0$. Then $T_{P_{k+2}} \Uparrow (m+1)(M_{k+1}) = T_{P_{k+2}}(T_{P_{k+2}} \Uparrow m(M_{k+1})) \cup M_{k+1}$ and $T_{P_{k+2}} \uparrow (m+1)(M_{k+1}) = T_{P_{k+2}}(T_{P_{k+2}} \uparrow m(M_{k+1})) \cup T_{P_{k+2}} \uparrow m(M_{k+1})$, and it is clear that $T_{P_{k+2}} \Uparrow (m+1)(M_{k+1}) \subseteq T_{P_{k+2}} \uparrow (m+1)(M_{k+1})$. For the reverse inclusion, we note that under our present hypotheses we have $T_{P_{k+2}} \uparrow (m+1)(M_{k+1}) = T_{P_{k+2}}(T_{P_{k+2}} \Uparrow m(M_{k+1})) \cup T_{P_{k+2}} \Uparrow m(M_{k+1})$ and so it suffices to show that $T_{P_{k+2}} \Uparrow m(M_{k+1}) \subseteq T_{P_{k+2}}(T_{P_{k+2}} \Uparrow m(M_{k+1})) \cup M_{k+1}$ or in other words that $I_{[k+2,m]} \subseteq T_{P(k+1)}(I_{[k+2,m]}) \cup I_{k+1}$. Since this latter set is equal to $I_{[k+2,m+1]}$ by the recursion equations of Corollary 4.6, the inclusion we want follows from the monotonicity of the sets $I_{[k+2,m]}$ relative to m. We conclude, therefore, that $T_{P_{k+2}} \Uparrow (m+1)(M_{k+1}) = T_{P_{k+2}} \uparrow (m+1)(M_{k+1})$.

Finally, $I_{[k+2,m+1]} = I_{k+1} \cup T_{P(k+1)}(I_{[k+2,m]}) = M_{k+1} \cup T_{P_{k+2}}(T_{P_{k+2}} \uparrow m(M_{k+1})) = M_{k+1} \cup T_{P_{k+2}}(T_{P_{k+2}} \Uparrow m(M_{k+1})) = T_{P_{k+2}} \Uparrow (m+1)(M_{k+1}) = T_{P_{k+2}} \uparrow (m+1)(M_{k+1})$, by the conclusions of the previous paragraph. Therefore, $I_{[k+2,m+1]} = T_{P_{k+2}} \uparrow (m+1)(M_{k+1})$. From this we obtain, by induction, the equality $I_{[k+2,m]} = T_{P_{k+2}} \uparrow m(M_{k+1})$ for all m and with it the equality $I_{k+2} = M_{k+2}$ as required. ∎

The details of the induction proof just given also establish the following proposition.

Proposition 4.12 Let $P = P_1 \cup \ldots \cup P_m$ be a stratified normal logic program. Then we have $T_{P_{k+1}} \Uparrow i(M_k) = T_{P_{k+1}} \uparrow i(M_k)$ for all i and $k = 0, \ldots, m-1$. ∎

Example 4.13 (1) Consider again the program in Example 3.13. We have already noted that the sequence of iterates $(T_P^n(\emptyset))$ does not converge in Q and that this program is stratified (with strata $P_1 = \{q(o) \leftarrow\}$ and $P_2 = \{p(s(o)) \leftarrow \neg q(o), p(x) \leftarrow r(x), r(x) \leftarrow p(x)\}$). A straightforward computation using the definitions made earlier in connection with stratified programs shows that $M_1 = T_{P_1} \uparrow \omega(\emptyset) = \{q(o)\}$ and that $M_P = M_2 = T_{P_2} \uparrow \omega(M_1) = \{q(o)\}$. On the other hand, the level mapping l given in the proof of Proposition 4.2 is, in this case, defined by $l(q(t)) = 0$ and $l(p(t)) = l(r(t)) = 1$ for all ground terms t. Thus, it turns out that $I_{[0,m]} = \{q(o)\}$ for all $m > 0$ so that $I_0 = \{q(o)\}$. Further straightforward computations show that $I_{[1,m]}$ also equals $\{q(o)\}$ for all m and hence that $I_{[P]} = I_1 = \{q(o)\} = M_P$ in accordance with Theorem 4.11.
(2) Consider the following program P:

$$q(o) \leftarrow$$

$$q(s^2(x)) \leftarrow q(x)$$

$$p(x) \leftarrow \neg q(x)$$

$$p(s^2(x)) \leftarrow \neg p(x)$$

$$p(x) \leftarrow p(x)$$

This program is not stratified but it is semi-strictly level-decreasing with respect to the level mapping l in which $l(q(s^n(o))) = 0$ and $l(p(s^n(o))) = n + 1$ for all n. In fact, I_0 is the set $\{q(s^{2n}(o)); n \in N\}$. Part 2 of Corollary 4.6 applies to the sub-program of P consisting of the set "definition of p remove the clause $p(x) \leftarrow p(x)$". This observation simplifies the computation of $I_{[P]}$ which in fact is the set $I_0 \cup \{p(s^n(o)); n \in N, n \text{ not a multiple of } 4\}$.

Note 4.14 For an arbitrary normal logic program P (whether stratified or not), let M^P denote $T_P \uparrow \omega(\emptyset)$, as defined earlier. By Lemma 4 of [1], M^P is a model for P. Thus:
(1) Apply this to the "even numbers" program, Example 3.12, which is not stratified. Then M^P is the set B_P, which is a model for P but is not a fixed point of T_P. Here of course $I_{[P]}$ is the set $\{p(s^{2n}(o)); n \in N\}$ of even numbers, and clearly $I_{[P]} \subset M^P$.
(2) For the Example 4.13 (2), which again is not stratified, M^P is the set $I_0 \cup \{p(s^n(o)); n \in N\}$. This is a fixed point of T_P, but is not minimal since $I_{[P]} \subset M^P$. Now partition P into "strata" $P = P_1 \cup P_2$, where $P_1 = \{q(o) \leftarrow, q(s^2(x)) \leftarrow q(x)\}$ and $P_2 = \{p(x) \leftarrow \neg q(x), p(s^2(x)) \leftarrow \neg p(x), p(x) \leftarrow p(x)\}$, and let $M_2 = T_{P_2} \uparrow \omega(M_1)$, where $M_1 = T_{P_1} \uparrow \omega(\emptyset)$, as defined earlier. Then M_2 is the set $I_0 \cup \{p(s^n(o)); n \in N, n \neq 0\}$ which is another fixed point of T_P, and we have $I_{[P]} \subset M_2 \subset M^P$.
(3) Taking P as in Example 4.13 (2) but removing the clause $p(x) \leftarrow p(x)$, we obtain that M^P is the set $I_0 \cup \{p(s^n(o)); n \in N\}$ and that M_2 is the set $I_0 \cup \{p(s^n(o)); n \in N, n \neq 0\}$. Both of these sets are models for P, but neither is a fixed point of T_P nor a minimal model. Indeed, the only fixed point of T_P is the set $I_0 \cup \{p(s^n(o)); n \in N, n \text{ not a multiple of } 4\}$. Of course, the uniqueness of the fixed point just noted is a consequence of the fact that P is in fact strictly level-decreasing with respect to an obvious level mapping.

4.2 The Case of Finite Level Mappings

As can be seen from Example 4.13 (2), the sets I_n defined in Construction 4.4 need not be finite, and this is true whether or not P is stratified. The question therefore arises as to whether or not it is possible to find a sequence (J_n) of finite sets J_n, possibly iterates of some I, which converges in Q to $I_{[P]}$. In particular, this question was prompted by the Prolog program written by Hitzler in [9] in order to calculate iterates and sequences of approximations, and which provided partial motivation for this study. To finish, we briefly record the facts which show that the answer to these questions is in the affirmative when P is semi-strictly level-decreasing with respect to a *finite* level mapping l and is also stratified by $P = P_1 \cup \ldots \cup P_m$, say. We make the following construction in which, in order to ease notation, we write $T_i^{[n]}$ in place of $(T_{P_i})_{[n]}$ for all i and n.

Construction 4.15 We construct the sequence (J_n) in I_P as follows: (i) P_1 is definite and $\mathcal{L}_n$ is finite for every n. Hence, for each n, the sequence $(T_1^{[n]} \Uparrow k(\emptyset))_{k \in N}$ is monotonic increasing with k and is, therefore, eventually constant with value $J_{n,1}$, say. (ii) By Lemma 10 of [1], we see that for each n the sequence $(T_{i+1}^{[n]} \Uparrow k(J_{n,i}))_{k \in N}$ is monotonic increasing with k, where $i = 1, \ldots, m-1$. Hence, it too is eventually constant with value $J_{n,i+1}$, say, on using the finiteness of the $\mathcal{L}_n$ again. Finally, we put $J_n = J_{n,m}$.

The proof of the following theorem may be found in [10].

Theorem 4.16 *Let P be a normal logic program which is stratified and is semi-strictly level-decreasing with respect to a finite level mapping l. Then the sequence $(J_n)_{n \in N}$ as defined in Construction 4.15 converges in Q to M_P.*

Remark 4.17 We close by comparing the complexities of the different approaches discussed in the present paper, at least for ω-level mappings.
(i) For strictly level-decreasing programs, it suffices to compute the sequence $(T_P^n(\emptyset))$ to obtain the unique supported model for the program, and therefore only a single limit is involved.
(ii) Construction 4.15 for programs which are stratified and semi-strictly level-decreasing with respect to a finite level mapping requires one to compute the single sequence (J_n). Moreover, each member of this sequence is itself obtained by a finite computation. Again, therefore, only a single limit is required in this case.
(iii) The approach of Apt, Blair and Walker [1] or the use of Construction 4.4 in the case of stratified programs requires the computation of the limits of finitely many sequences $(T_{P_{k+1}} \uparrow n(M_k))$.
(iv) Using Construction 4.4 for semi-strictly level-decreasing programs involves the computation of the limit of the sequence (I_n), where each I_n is itself obtained by constructing the sequence $(I_{[n,m]})_m$ and its limit. So, in this case, at most countably many limits have to be computed. If the program is semi-strictly level-decreasing with respect to a finite level mapping, the sequence $(I_{[n,m]})_m$ stabilizes after finitely many steps, and therefore only a single limit needs to be computed.

References

[1] K.R. Apt, H.A. Blair and A. Walker, Towards a Theory of Declarative Knowledge. In: Jack Minker (Ed.), Foundations of Deductive Databases and Logic Programming. Morgan Kaufmann Publishers Inc., Los Altos, 1988, pp. 89-148.

[2] K.R. Apt and D. Pedreschi, Reasoning About Termination of Pure Prolog Programs, *Information and Computation* **106** (1) (1993), 109-157.

[3] A. Batarekh, Convergence Issues and Fixed Points of Non-Monotone Maps. In: E.L. Lusk and R.A. Overbeek (Eds.), Proceedings of the North American Conference on Logic Programming. MIT Press, 1989, pp. 257-273.

[4] A. Batarekh, Fixpoint Techniques for Non-Monotone Maps. In: H. Kirchner and W. Wechler (Eds.), Proceedings of the Second International Conference on Algebraic and Logic Programming, Nancy, France, 1990. Lecture Notes in Computer Science Vol. 463, Springer-Verlag, pp. 132-143.

[5] M. Bezem, Characterizing Termination of Logic Programs with Level Mappings. In: E.L. Lusk and R.A. Overbeek (Eds.), Proceedings of the North American Conference on Logic Programming. MIT Press, 1989, pp. 69-80.

[6] A. Edalat, Dynamical Systems, Measures and Fractals via Domain Theory, *Information and Computation* **120** (1) (1995), 32-48.

[7] M. Fitting, Logic Programming Semantics using a Compact Data Structure. In: Z.W. Ras and M. Zemankova (Eds.), Proceedings of the ACM SIGART International Symposium on Methodologies for Intelligent Systems, 1986, pp. 247-255.

[8] M. Fitting, Metric Methods: Three Examples and a Theorem, *J. Logic Programming* **21** (3) (1994), 113-127.

[9] P. Hitzler, Approximation von Fixpunkten normaler logischer Programme, Studienarbeit in Informatik, Universität Tübingen, 1997.

[10] P. Hitzler, Topology and Logic Programming Semantics, Diplomarbeit in Mathematik, Universität Tübingen, 1998.

[11] M.A. Khamsi, V. Kreinovich and D. Misane, A New Method of Proving the Existence of Answer Sets for Disjunctive Logic Programs: A Metric Fixed Point Theorem For Multi-Valued Mappings. Proceedings of the Workshop on Logic Programming with Incomplete Information, Vancouver, B.C., Canada, October 1993, pp. 58-73. [Also obtainable as a Technical Report via FTP from the Department of Computer Science, University of Texas at El Paso, 1994].

[12] J.W. Lloyd, Foundations of Logic Programming. Springer-Verlag, 2nd Edition, Berlin, 1988.

[13] M.J. Maher, Equivalences of Logic Programs. In: Jack Minker (Ed.), Foundations of Deductive Databases and Logic Programming. Morgan Kaufmann Publishers Inc., Los Altos, 1988, pp. 627-658.

[14] S. Priess-Crampe and P. Ribenboim, Fixed Points, Combs and Generalized Power Series, *Abh. Math. Sem. Univ. Hamburg* **63** (1993), 227-244.

[15] S. Priess-Crampe and P. Ribenboim, Ultrametric Spaces and Logic Programming. Preprint, October 1997, pp. 1-12.

[16] T. Przymusinski, On the Declarative Semantics of Deductive Databases and Logic Programs. In: Jack Minker (Ed.), Foundations of Deductive Databases and Logic Programming. Morgan Kaufmann Publishers Inc., Los Altos, 1988, pp. 193-216.

[17] T. Przymusinski, Well-Founded and Stationary Models of Logic Programs, *Annals of Mathematics and Artificial Intelligence* **12** (1994), 141-187.

[18] T. Przymusinski, Semantics of Normal and Disjunctive Logic Programs: A Unifying Framework. In: Proceedings of the Workshop on Non-Monotonic Extensions of Logic Programming at the 11th International Logic Programming Conference, ICLP' 95, Santa Margherita Ligure, Italy, June 1994. Springer-Verlag, 1995, pp. 43-67.

[19] A.K. Seda, Topology and the Semantics of Logic Programs, *Fundamenta Informaticae* **24** (4) (1995), 359-386.

[20] A.K. Seda, Quasi-metrics and the Semantics of Logic Programs, *Fundamenta Informaticae* **29** (1) (1997), 97-117.

[21] A.K. Seda and P. Hitzler, Strictly Level-Decreasing Logic Programs. Preprint, February 1998, pp. 1-20.

[22] M.B. Smyth, Totally bounded spaces and compact ordered spaces as domains of computation. In: G.M. Reed, A.W. Roscoe and R.F. Wachter (Eds.), Topology and Category Theory in Computer Science. Oxford University Press, 1991, pp. 207-229.

[23] V. Stoltenberg-Hansen, I. Lindström and E.R. Griffor, Mathematical Theory of Domains. Cambridge Tracts in Theoretical Computer Science No. 22, Cambridge University Press, Cambridge, 1994.

[24] A. Van Gelder, Negation as Failure Using Tight Derivations for General Logic Programs. In: Jack Minker (Ed.), Foundations of Deductive Databases and Logic Programming. Morgan Kaufmann Publishers Inc., Los Altos, 1988, pp. 149-176.