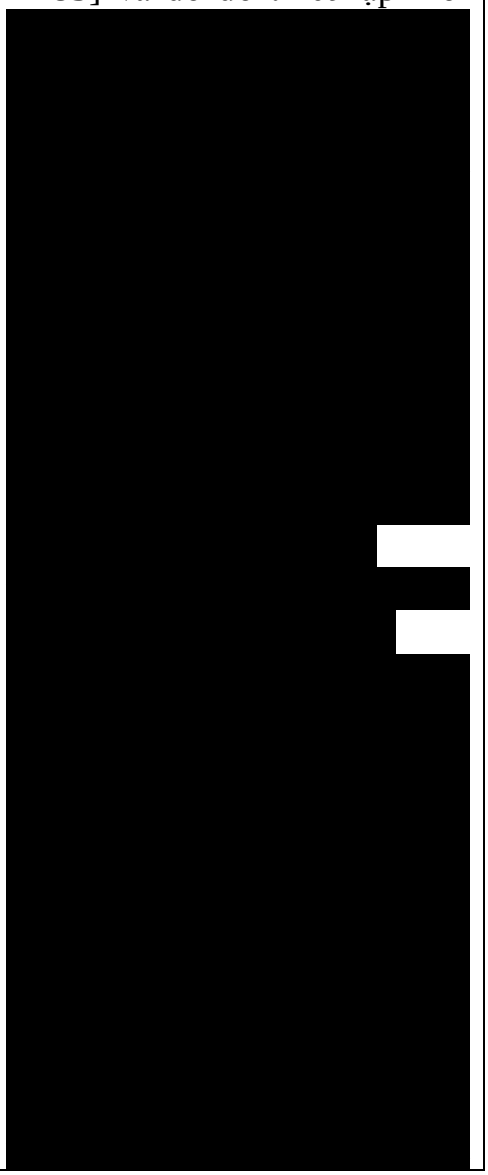
| | |
|---|---|
| Weakly Stratified Logic Programs<br>Contents<br>1 Introduction 1<br>2 Notation and Definitions 3<br>3 Weakly Stratified | Chương trình logic phân tầng yếu<br>Nội dung<br>1 Giới thiệu<br>2 Kí hiệu và định nghĩa<br><br>3 Các chương trình phân tầng |

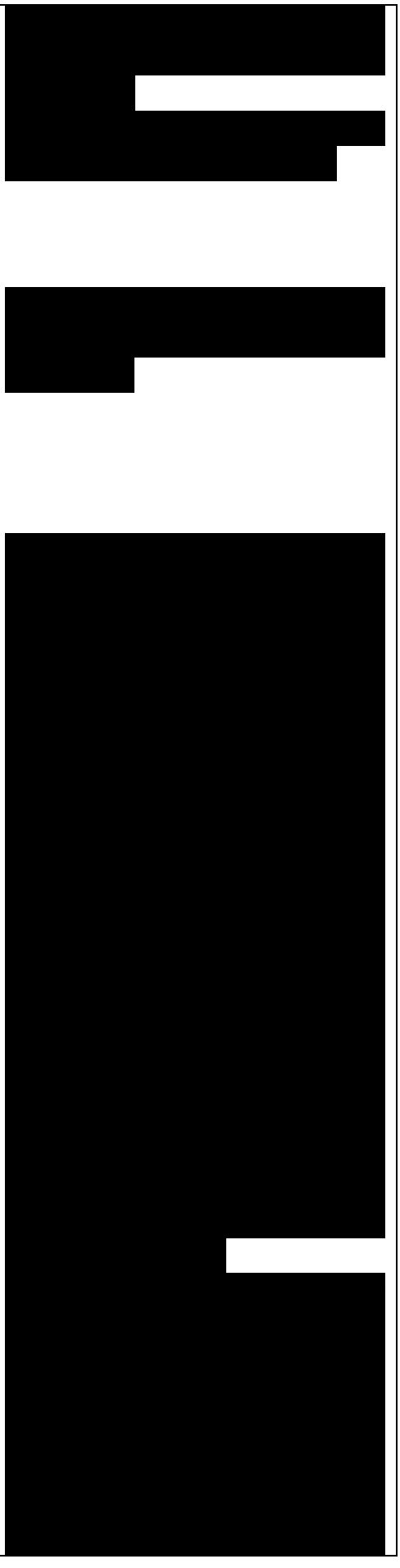
| | |
|---|---|
| Programs and Weakly Perfect Models 4 | yếu và các mô hình hoàn hảo yếu |
| 4 Properties of Weakly Perfect Models 10 | 4 Các tính chất của mô hình hoàn hảo yếu |
| 5 Relation to Other Proposed Semantics 14 | 5 Mối quan hệ với các ngữ nghĩa được đề xuất khác |
| | declarative semantics: một số tài liệu dịch là ngữ nghĩa mô tả |
| 1 Introduction | 1 Giới thiệu |
| In this paper we address the problem of declarative (or intended) semantics for logic programs. In [ABW88] Apt, Blair and Walker and - independently - Van Gelder [VG89] (see also [CH85]) introduced the class of stratified logic programs, defined a unique 'natural' Herbrand model Mp of a stratified logic program and argued that this model should be used to represent the intended meaning of such programs. | Trong bài báo này, chúng tôi phân tích vấn đề ngữ nghĩa khai báo (chủ định) cho các chương trình logic. Trong [ABW88] Apt, Blair và Walker độc lập với Van Gelder [VG89] (xem thêm [CH85]) cũng giới thiệu một lớp các chương trình logic phân tầng, định nghĩa mô hình Herbrand "tự nhiên" duy nhất của chương trình logic phân tầng và cho rằng mô hình này sẽ được dùng để biểu diễn ý nghĩa chủ định của các chương trình đó. |
| | intended meaning: một số tài liệu dịch là ý niệm đã định |
| T.Przymusinski [Prz88a] extended the class of stratified logic programs to a wider class of locally stratified programs and introduced the notion of a perfect model of a logic program. He showed that every locally stratified logic program has exactly one perfect Herbrand model, which - for stratified programs - coin¬cides with the 'natural' model Mp. In [Prz89b] the definition of a perfect model was extended to the class of non-Herbrand models. | T.Przymusinski [Prz88a] đã mở rộng lớp chương trình logic phân tầng thành lớp chương trình phân tầng cục bộ rộng hơn và đưa vào khái niệm mô hình hoàn hảo của một chương trình logic. Ông ta đã chứng tỏ rằng mỗi chương trình logic phân tầng cục bộ có đúng một mô hình Herbrand hoàn hảo, đối với các chương trình phân tầng, mô hình này trùng với mô hình "tự nhiên" Mp. Trong [Prz89b], định nghĩa mô hình hoàn hảo được mở rộng sang lớp mô hình phi |

| | |
|---|---|
| | Herbrand. |
| The class of perfect models of a logic program has many natural and desirable properties. All perfect models are minimal and for positive logic programs the notions of a minimal and perfect model coincide. As shown in [Prz88a, Prz89b] (see also [Lif88, Prz88b]), perfect models are models of McCarthy's prioritized circumscription [McC80, McC86, Lif85] and thus closely relate the semantics of logic programs to an important formalization of non-monotonic reasoning. Moreover, for stratified programs T. Przymusinski defined in [Prz89b] the so called SLS-resolution, which generalizes the standard SLD-resolution and is sound and complete with respect to the perfect model semantics. | Lớp mô hình hoàn hảo của một chương trình logic có các tính chất tự nhiên và tính chất đáng quan tâm. Tất cả các mô hình hoàn hảo đều cực tiểu và đối với các chương trình logic dương, khái niệm cực tiểu và mô hình hoàn hảo trùng nhau. Như đã trình bày trong [Prz88a, Prz89b] (cũng như [Lif88, Prz88b]), mô hình hoàn hảo là các mô hình hạn chế phạm vi ưu tiên của McCarthy [McC80, McC86, Lif85] và do đó thiết lập mối |
| The class of logic programs with well-defined perfect model semantics is fairly broad and is not limited to the class of stratified or locally stratified logic programs (see Example 3.4). Recently, however, several researchers pointed out that there exist interesting and useful logic programs with a natural intended semantics, which do |  |

not have perfect models [GL88, VGRS90].

Example 1.1 Consider the program P [GL88, Example (2)] given by:
p(l,2) <-
q(X) <- p(X,Y), -<q(Y).
After instantiating, P takes the form:
p(l,2) <-
q{l) <- p(l,2),^q(2)
q{ 1) «- p(l,l),^g(l)
q(2) <- p(2,2), _>g(2)
q( 2) <- p(2,l),-g(l).
This program is not locally stratified, because the priority relation < between atoms - as determined by the dependency graph of P - is not a partial order (see [Prz88a]), namely g(l) < q{2) and q(2) < g(l). Moreover, one can easily verify that P does not have perfect models. On the other hand, it seems clear that the intended semantics of P is well-defined and is characterized by the Herbrand model M = {p(l, 2), g(l), -ig(2), -ip(l, 1), ->p(2,2),-<p(2,1)} of P. The same results would be produced by Prolog, which further confirms our intuition.

The cause of this problem is fairly clear. Program P appears to be seman¬tically equivalent to a locally stratified program P* consisting only of clauses (1) and (2). Clauses (3), (4) and (5) seem to be entirely



irrelevant, because p(l, l),p(2,1) and p(2,2) can be assumed false in P. At the same time, they are the ones that destroy local stratifiability of P and cause the non-existence of perfect models. □

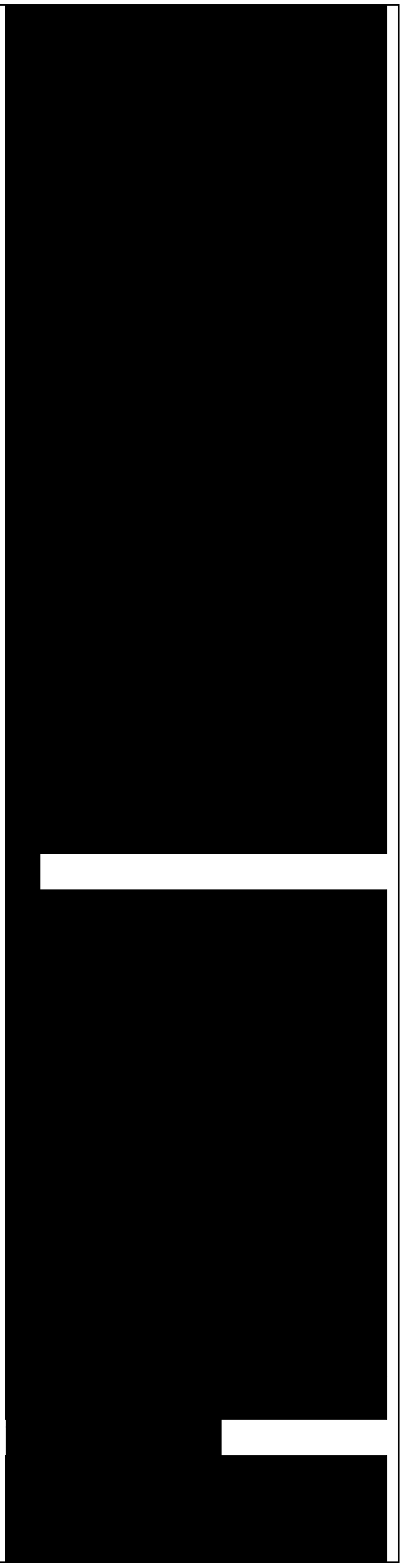
In this paper we propose a natural extension of the class of (locally) strat¬ified programs to a broader class of weakly stratified programs, which includes programs of the type discussed above. We also introduce the class of weakly per¬fect models of logic programs. Every weakly stratified program has a (unique) weakly perfect model, but the class of programs admitting weakly perfect mod¬els is significantly larger than the class of weakly stratified programs. We prove that if a logic program has both a perfect model and a weakly perfect model then they must coincide. Therefore, the weakly perfect model semantics is fully compatible with the perfect model semantics.

The main idea behind those concepts is to remove 'irrelevant' relations in the dependency graph of a logic program and to substitute components of the dependency graph for its vertices in the definitions of stratification and perfect models. In the proposed

extension we stick very closely to the original definitions of stratified programs and their models. We decompose the program into strata and base its semantics on the iterated least model approach, which can also be thought of as the least fixed point of a suitable operator. However, the decomposition into strata is performed dynamically rather than statically. In the case of a single stratum the weak perfect semantics is simply equivalent to the least model semantics. As opposed to the well-founded semantics [VGRS90, Prz89a], the ¿east models considered are always 2-valued.

As a result we obtain a broader class of programs, whose semantics possesses the same natural features as the perfect model semantics. In particular, weakly perfect models are minimal models of P and they coincide with models of prior¬itized circumscription. Moreover, for weakly stratified programs SLS-resolution is still sound and complete with respect to the weakly perfect model semantics (for non-floundering queries).

The class of programs with weakly perfect models is different from the class of



programs with well-founded semantics - proposed by Van Gelder, Ross and Schlipf in [VGRS90] - and from the class of programs with unique stable model semantics - defined by Gelfond and Lifschitz in [GL88]. However, the three approaches are closely related. In particular, the class of weakly stratified logic programs is contained in the class of programs with 2-valued well-founded models and the latter class is contained in the class of programs with unique stable models [VGRS90].

2 Notation and Definitions

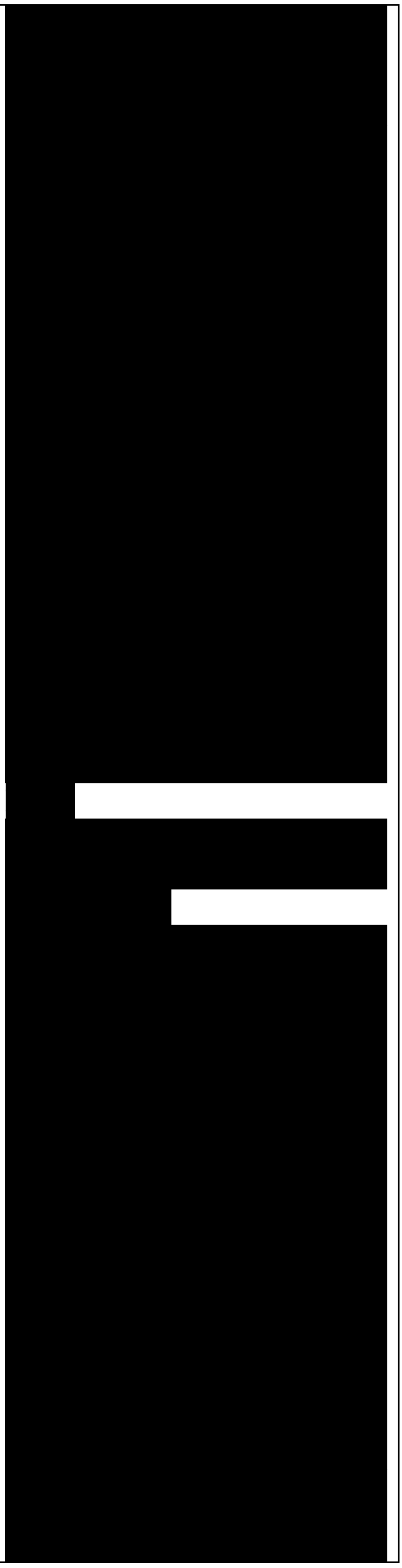
By a logic program, we mean a finite set of universally quantified clauses of the form
V (A 4— L\ A ... A Lm),

where m>0, A is an atom and L, are positive or negative literals (see [Llo84]). Following a standard convention, such clauses will be simply written as:
A i— Li,..., Lm.

The alphabet of a program P consists of all the constant, predicate and function symbols that appear in P, a countably infinite set of variable symbols, connectives (-i,A,V), quantifiers (3, V) and the usual punctuation symbols. We assume that if P does not contain any constants, then

one is added to the alphabet. The language L of P consists of all the well-formed formulae of the so obtained first order theory. The Herbrand base Hp of P is the set of all ground atoms of the theory. Throughout the paper, instead of the program P we consider its instantiated version ground(P), which, in the presence of function symbols, may be infinite. We consider only Herbrand models of P, but a suitable extension to arbitrary models can be obtained in a straightforward way, by following the approach presented in [Prz89b], where non-Herbrand perfect models of logic programs are introduced.

By a partial interpretation M of P we mean a signed subset of the Herbrand base Hp of P, i.e., a subset of Hp, some of whose elements may be negated and thus considered negative. An interpretation of P is a partial interpretation, which decides the truth or falsity of all atoms from Hp. A model M of P is an interpretation of P, in which all clauses from P are satisfied. A model M of P is the (unique) least model of P, if it contains less positive atoms than any other model M' of P.



Definition 2.1 (Dependency Graph [ABW88, VG89]) The vertices of the dependency graph Gp of a program P are all ground atoms appearing in P. The edges of Gp are defined as follows. For every clause

A <- Ai,..., An, •/?!,..., •/?,,,.

in P and for every i <n a positive directed edge from A, to A is included in Gp and for every j < ma negative directed edge from Bj to A is included in Gp. The dependency (or priority) relations < and < between ground atoms of P are defined by:

(i) A < B iff there is a directed path from A to B;

(ii) A < B iff there is a directed path from A to B passing through a negative edge. □

3 Weakly Stratified Programs and Weakly Per¬fect Models

In this section we extend the class of (locally) stratified programs to a broader class of weakly stratified programs and we introduce the class of weakly perfect models of logic programs. The underlying idea is to introduce the components of the graph Gp and to substitute them for the vertices in the definition of stratification and perfect models. First, we define the notion of a component.

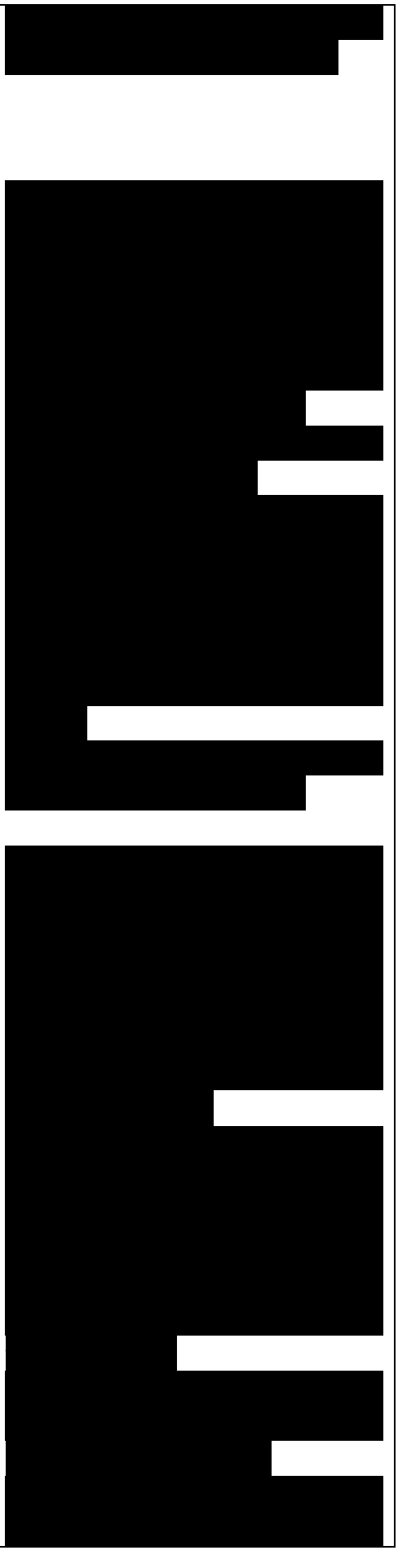
Definition 3.1 Let ~ be the equivalence relation between ground atoms of P defined as follows:
.4 ~ — (.4 = B) V (A < B A B < A).
We will call its equivalence classes components of Gp. A component is trivial if it consists of a single element A, such that A ft A. □

According to the above definition, two distinct ground atoms A and B are equivalent if they are related by mutual negative recursion, i.e. recursion passing through negative literals. Mutual negative recursion is the primary cause of difficulties with a proper definition of declarative semantics of logic programs.
We now introduce an order relation -< between the components of the dependency graph Gp.
Definition 3.2 Let C\ and C2 be two components of Gp. We define:
C1<C2 = C1^C2A 3Ax g e,. 3.4-2 e C, (Ax < .4,).
We call a component C\ minimal, if there is no component C2 such that C2 -< Gi. □
Clearly, the relation -< and therefore the minimal components of Gp are completely determined by the syntactic form of the program P. It is easy to see that it does not matter which atoms A\ and

A2 one chooses from the respective components. More precisely:
C1<C2 = C1^C2A VAx g Ci,VA2 g C2 (Ax < A2).
The order relation -< between the components of the dependency graph Gp corresponds to the dependency relation < between its vertices, but, as opposed to <, it has the added advantage of always being a partial order.

Definition 3.3 By a partial order we mean an asymmetric and transitive re¬lation. An ordering < is said to be well-founded if it does not contain infinite decreasing sequences Aq > A\ > A2 > ... . □
Proposition 3.1 The relation -< is always a partial order.

Proof: The relation -< is asymmetric, because if C -< C' and C' -< C, then for any .4 G C and A' G C' we have A < A' and A' < A. This implies that A and A' belong to the same component, which contradicts C ± C'.
The relation -< is transitive, because if C -< C' -< C", then for any A G C, A' G C", A" G C" we have A < A' < A". By transitivity of < and asymmetry of this implies that C <C". □
The following proposition immediately follows from [Prz88a, Theorem 5.4].
Proposition 3.2 For a logic program p the following are



equivalent:
(i) p is locally stratified;
(ii) The dependency relation < is a well-founded partial order;
(Ui) AU components of Gp are trivial and the partial ordering of components -< is well-founded. □

Naturally, if the (instantiated) program is finite, then the condition of well-foundedness in (iii) is automatically satisfied and therefore p is locally stratified if and only if all components of Gp are trivial.

The ordering -< has the following simple, but important, property: for any two distinct components C,C', of which at least one is non-trivial, either they are independent of each other, in the sense that there are no A £ c and B £ C', which are related by A < B or B < A, or c -< C', or c' -< c (but not both, due to the fact that -< is a partial ordering).

Proposition 3.3 Suppose that c and C' are distinct components not both of which are trivial. Suppose also that A G c, B G C' and A < B. Then c <C'.

Proof: Suppose e.g. that c is not trivial. There is an A' in c such that A' < A and therefore A' < B which implies c <σ. □

The above proposition implies

that if c and Ċ are distinct components - not both trivial - and if c, C' are not related by either c -< Ċ or C' -< c, then models of p restricted to those components are completely independent of each other, because they involve predicates, which are not related by the dependency relation.

For any logic program p we introduce the following definitions.

Definition 3.4 By the bottom stratum S(P) of p we mean the union of all minimal components of p, i.e.

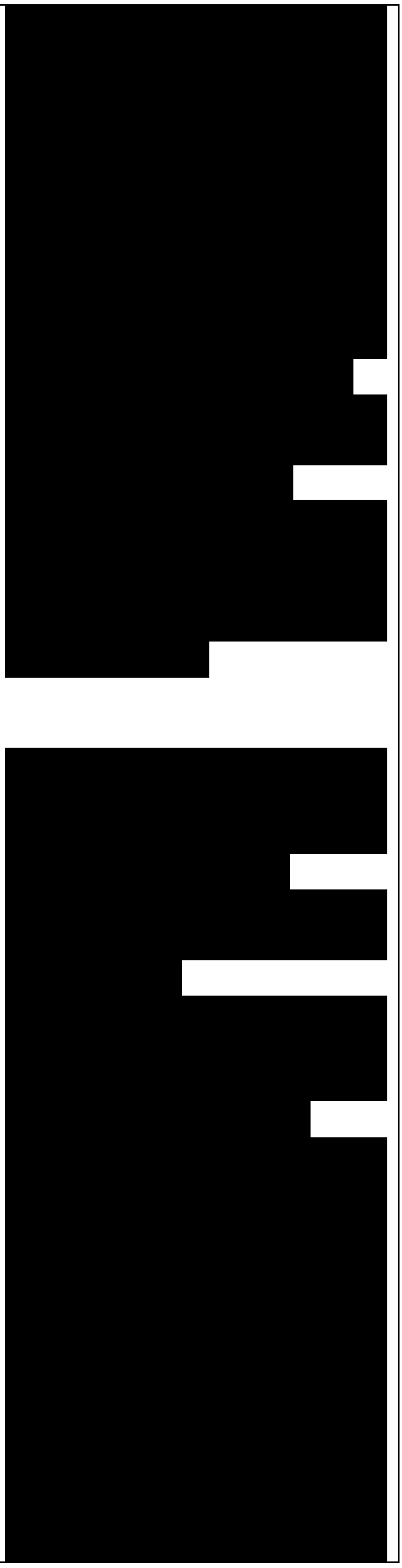
S(P) = [_J{C : c is a minimal component of Gp}.

By the bottom layer L(P) of p we mean the corresponding subprogram of p, i.e.

L(P) = 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). □

Observe that, if the instantiated program is infinite, then it may not have any minimal components and thus its bottom stratum may be empty. For example, the bottom stratum of the program p(X) -<p(f(X)) is empty. In view of Proposition 3.3, non-trivial components c in the bottom stratum of p are completely independent of the



remaining components in the bottom stratum.

Example 3.1 Consider the program P from Example 1.1. The dependency ordering is given by the following relations:
q( 1) < q(2), q(2) < q(l), q(l)>p(l,2), q(l) > p(l, 1), q(2) > p(2,2), q(2) > p(2,1).
Program P has five components:C% = {g(l), q{2)}, C2 = {p(l,2)},C3 = {p(l, 1)}, C4= {p{2,2)} and C5 = {p{2,1)}. Clearly, Cu -< C\, for any 2 < k < 5. Consequently, the bottom stratum S(P) of P - defined as the union of minimal components of P - is given by:
S(P) = {p(l, 2),p(l, l),p(2,2),p{2,1)},
and the bottom layer L(P) of P, i.e. the set of all clauses from P whose heads belong to S(P), is:
L(P) = {p( 1,2) <-}.
Observe, that the bottom layer L(P) of the above program P has the least (Herbrand) model M = {p(l,2),-ip(l,l),-ip(2,2),-ip(2,l)}. □
If the bottom layer L(P) of P has the least model M then we can use it to remove from P all 'irrelevant' clauses and literals. More generally, we now introduce the operation of reduction, which reduces a given program P modulo its

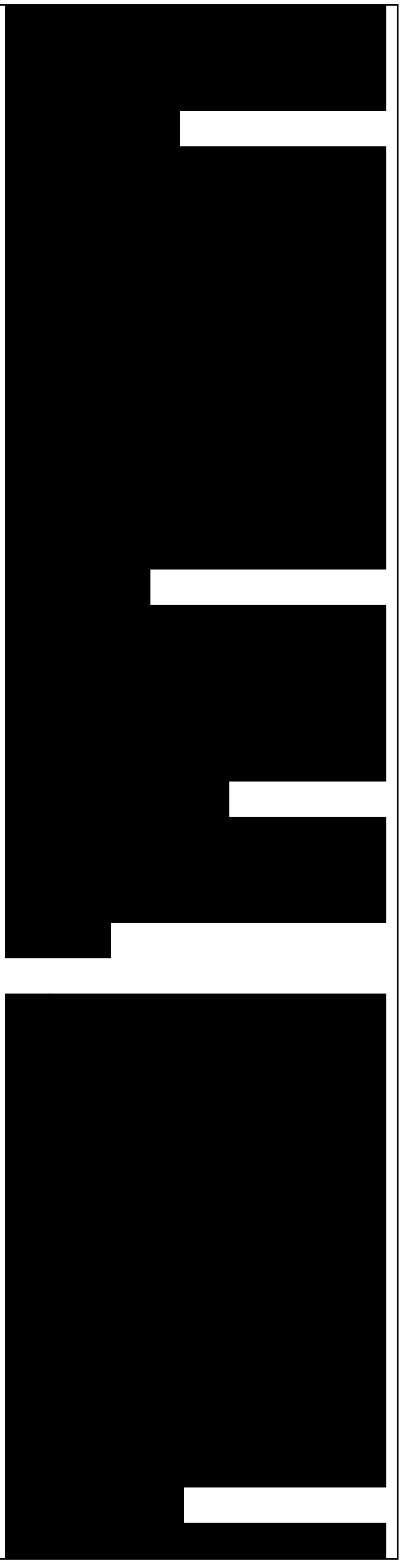
partial interpretation M, by essentially applying the Davis-Putnam rule to P [CL73].

Definition 3.5 Let P be a logic program and let M be a partial interpretation of P. For an atom, A in Hp, we will say that M \= A if A is in M and we will say that M |= ->A if -<A is in M. By a reduction of P modulo M we mean a new program obtained from P by performing the following two reductions:

- removing from P all clauses which contain a premise L such that M |= -■ L or whose head belongs to M (in other words, removing all clauses true in M);
- removing from all the remaining clauses those premises L which are satisfied in M, i.e. such that M \= L.

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

The so reduced program ^



does not contain any (positive or negative) atoms which occurred in M. In the Example 3.1 the reduced program P' = consists only of the clause:
q{ 1) <- -g(2).
Clearly, P' does not contain any literals from S(P). Observe that in the reduced program we got rid of all the 'irrelevant' clauses (3), (4) and (5) in P.
The idea behind the construction of weakly perfect models is as follows. Take any program P = P0 and let Mo = 0. Let P\ = . find the least model
M\ of the bottom layer L{P\) of Pi and reduce P modulo Mo U Mi obtaining a new program P2 = MJjMl • Find its bottom layer L{P2) and its least model M2 and let P3 = MoUm1um2 ' Continue the process until either the resulting k-th program Pf. is empty, in which case Mp = Mi U ... U Mf.-i is the weakly perfect model of P, or, otherwise, until either S(Pu) is empty or L{Pf.) does not have a least model, in which case Mp = Mi U ... U Mf.-i is the partial weakly perfect model of P.
We now generalize the above approach, by giving a formal transfinite defi¬nition of a (possibly, partial) weakly perfect model Mp of a logic program P.

Definition 3.6 Suppose that P

is a logic program, and let Pq = P, Mo = 0. Suppose that a > 0 is a countable ordinal such that programs P$ and partial interpretations Mg have been already defined for all 5 < a. Let

*„ = U Ms,
0<S<a

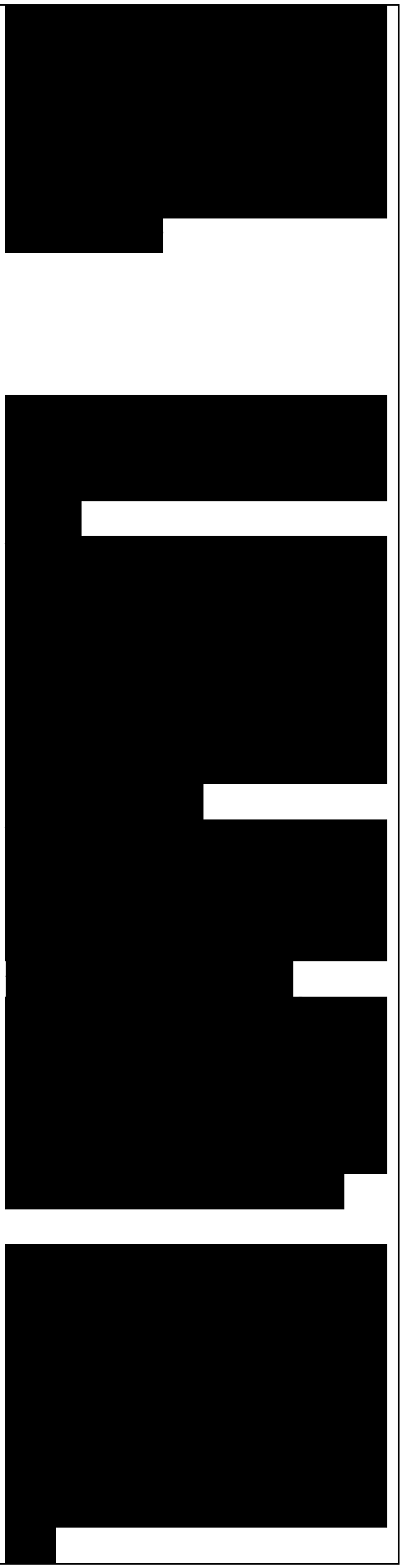
Pa = sa = S(Pa), La = L(Pa).

i\a

- If the program Pa is empty, then the construction stops and Mp = Na is the weakly perfect model of P;
- Otherwise, if the bottom stratum Sa = S(Pa) of Pa is empty or if the least model of the bottom layer La = L(Pa) of Pa does not exist, then the construction also stops and Mp = Na is the partial weakly perfect model of P.
- Otherwise, the partial interpretation Ma is defined as the least model of the bottom layer La = L(Pa) of Pa and the construction continues.

In the first two cases , a is called the breadth of P and is denoted by S(P). For 0 < a < S(P), the set Sa is called the a-th stratum of P and the program La is called the a-th layer of P.

In the process of constructing the strata Sa, some ground atoms may be eliminated 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 Mp. □



The construction always stops after countably many steps and therefore the (partial) weakly perfect model Mp of a program P is always defined and unique. A particularly important case of the above definition occurs when all the strata Sa consist only of trivial components or - equivalently - when all the program layers La are positive logic programs.

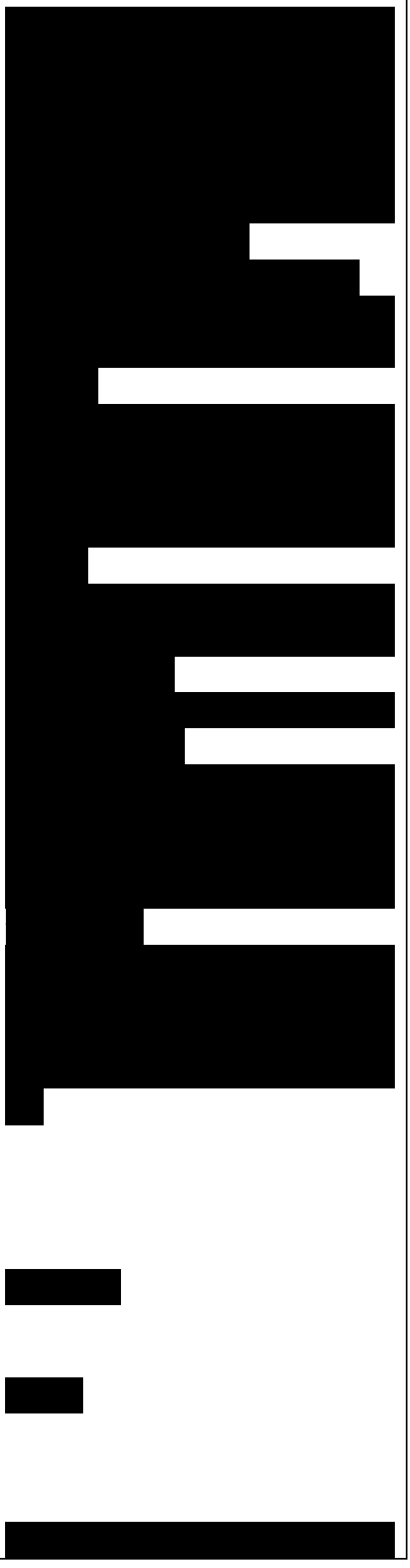
Definition 3.7 We say that a logic program P is weakly stratified if it has a weakly perfect model and if all of its strata Sa consist only of trivial components or - equivalently - when all of its layers La are positive logic programs. In this case, we call the set of program's strata {Sa : 0 < a < S(P)} the weak stratification of P. □

Remark 3.1 Observe, that since every positive logic program has the least model, a program P is weakly stratified if and only if whenever Pa is non-empty Sa = S(Pa) is also non-empty and consists only of trivial components.

Example 3.2 Consider the program P from Example 3.1. We obtain:
Pi = P, Si = S(P) = {p(l,2),p(l,l),p(2,2),p(2,l)},
Li=L(P) = {p(l,2)<-}.
and therefore
Mi ={p(l,2),-p(l,l),-p(2,2),-

p(2,l)}.
Consequently, P2 = = {g(l) <— ->g(2)}, S2 = = {(7(2)} is the union of minimal components of P2 and L2 = L{P2) = 0 is the set of clauses from P2 whose heads belong to S2. Therefore, M2 = {->g(2)}. As a result,
P3 = A/1 'u ;\12 = {g(1) 53 = {g(1)}' L3 = P3 End M* = 53 = {g(1)}'
Since P4 = MlUM2UM3 = the construction is completed, P is weakly stratified, {S1.S2.S3} *s 'ts weak stratification and
Mr = A/, U A/, U A/3 = {p(l, 2),g(l), -p(l, 1), -p(2,2), -p(2,1), -g(2)}

is its unique weakly perfect model. □
The preceding example can be easily modified to illustrate the notion of a partial weakly perfect model of a logic program.
Example 3.3 Consider the program from Example 1.1 with an added atomic fact p(2,1) For the new program P we have:
Pi =P, Si =S(P) ={p(l,2),p(l,l),p(2,2),p(2,l)},
Li = L{P) = {p(l, 2) 4-, p(2,1) ^-},
and therefore
Mi = {p(l, 2),p(2,1), ->p(2,2), ->p(l, 1)}.
Consequently,
P* = §; = {«(!) -«(2). «(2) <52 — {<7(1)><7(2)}> L2 — P2-
Since 1*2 does not have the



least model, the construction stops here and we obtain a partial weakly perfect model MP = {p(l,2),p(2,1),2), ->p(l, 1)}. □

The class of programs admitting weakly perfect models is much broader than the class of weakly stratified programs.

Example 3.4 Let P consist of clauses:

P 4- <1 i— -1P■

Then P has a single component and therefore its weakly perfect model is the least model of P, namely Mp = {p, ->g} (see Corollary 4.6). Clearly, P is not weakly stratified. See [PP88, Example 3.4] for a discussion of this example.

Example 3.5 Let P be as follows:

p i— q,^r q i— r,^p r i— p,^q

Again, P has a single component and therefore its weakly perfect model is the least model of P and thus it is empty. Naturally, P is not weakly stratified.

## 4 Properties of Weakly Perfect Models

The class of weakly stratified programs extends the class of (locally) stratified programs.

Theorem 4.1 Every (locally) stratified program is weakly stratified.

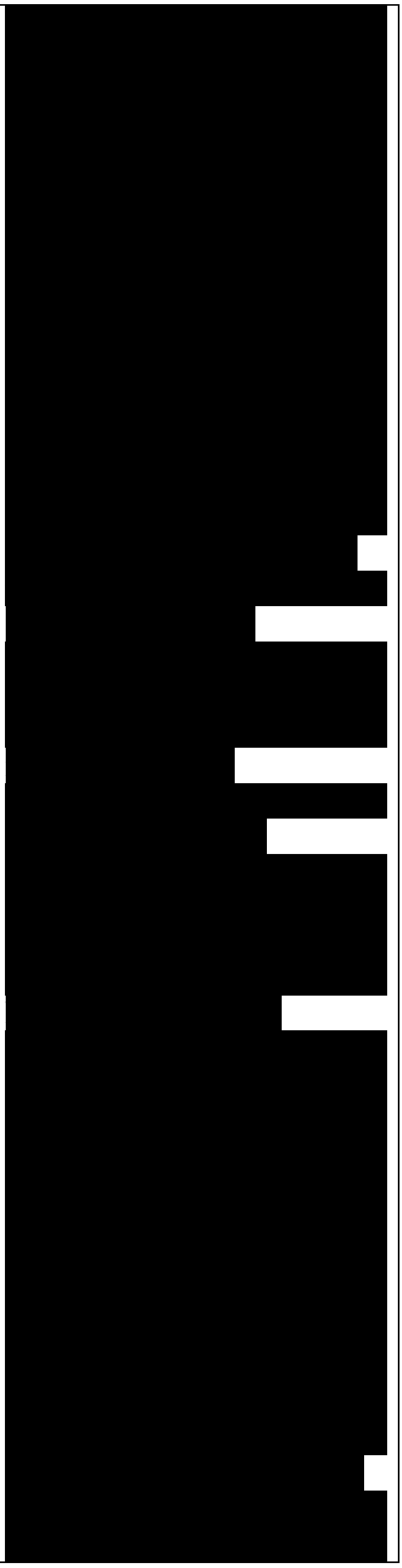
Proof: First, observe that if P is locally stratified, then so is the reduced program P' = for any partial interpretation M.

Therefore, for every a, the program Pa is also locally stratified. By Remark 3.1, it suffices to show that Sa = S(Pa) is not empty and consists only of trivial components, whenever Pa is non-empty. By Proposition 3.2, all components of Pa are trivial and their ordering -< is well-founded. Therefore, Sa = S(Pa) consists only of trivial components and is not empty, as long as Pa is non-empty. This completes the proof. □

Weakly perfect models share the property of minimality with perfect models.

Theorem 4.2 Every weakly perfect model is minimal.
Proof: Suppose that there is a weakly perfect model M of a program P, which is not minimal. Therefore, there exists a smaller model N of P. Let a < 5(P) be the smallest ordinal such that M\Sa ^ N\Sa, where by M\S we denote the model M restricted to the subset S of the Herbrand base Hp of P. By definition, M\Sa is the least model of La. Since M|S7 = iV|S7, for every 7 < a, it is easy to see that N\Sa must also be a model of La, which is strictly smaller than M\Sa. This is a contradiction. □

The existence of a perfect model of a logic program does



not always imply the existence of its weakly perfect model.

Example 4.1 Suppose that P contains clauses:
P(X) <- -p(/PO), p(x)^p(f(x)).

This program has the least model in which p(X) is true for every X and there¬fore it has a perfect model. It does not have a weakly perfect model because, although the program itself is not empty, its bottom stratum is empty.
As the following theorem shows, for a program P with a perfect model, the emptiness of bottom strata of its non-empty subprograms P' (or the non-well- foundedness of the ordering -< of components of P') can be the only cause of the non-existence of the weakly perfect model of P.

Theorem 4.3 Suppose that a logic program P has a perfect model. If all non¬empty subprograms Pa constructed in Definition 3.6 have non-empty bottom strata Sa = S(Pa) then P also has a weakly perfect model and the two models coincide.

Proof: Let K be a perfect model of P. We will prove, by induction on a, that the bottom layer La of the program Pa has the least model Ma and that this model coincides with the restriction Ka = K\Sa of the perfect model K to the a's

stratum Sa. This will show that the construction of the weakly perfect model described in Definition 3.6 can be successfully carried out to the end and that it results in a model, which coincides with the perfect model K.

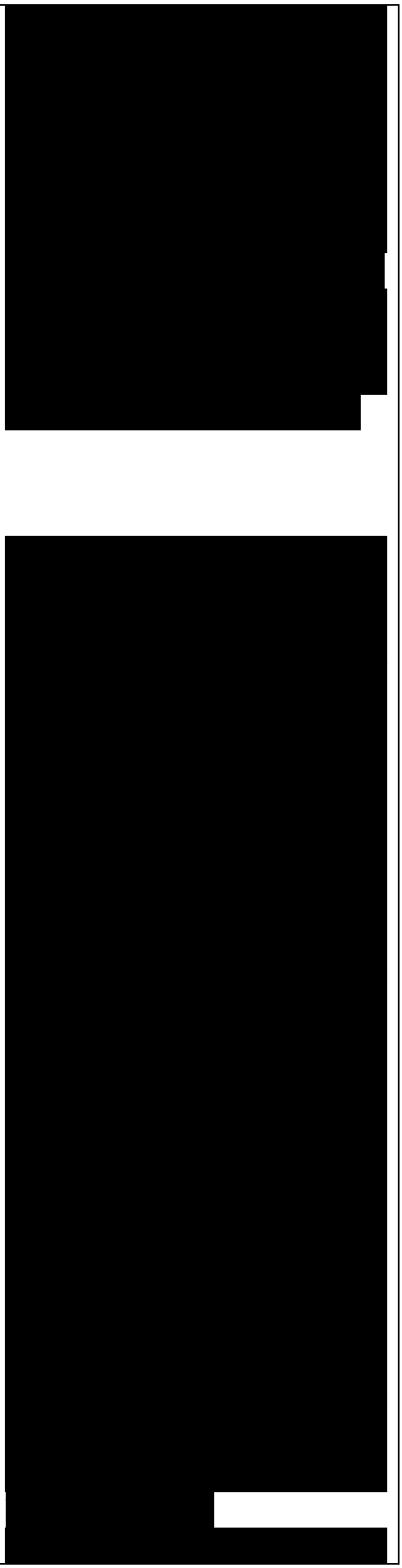
Suppose that our claim has been proven for all 7 < a. We will prove it for a. Let

*,, = U Ms,

0<S<a

Pa = ^~, Sa= S(Pa), La = L(Pa). i\a

By the assumption, if Pa is not empty, then so is Sa. Let K' be the restriction of the perfect model K to the set Hp — Uocic« First we show that K' is a perfect model of Pa. Let U' be any clause from Pa. There exists a clause U in P, such that U' can be obtained from U by removing from it some premises which are satisfied in Na. By the inductive assumption, these premises are also satisfied in K. Since K is a model of U, it follows that K' is a model of U', which implies that K' is a model of Pa. If K' were not perfect, then there would exist another model K" of Pa, which is preferred to K' (see [Prz89b]). Let K* be the union of Na and K". It is easy to see that K* is a model of P, which is preferred to K, which is impossible. This proves that K' is a perfect model of Pa,

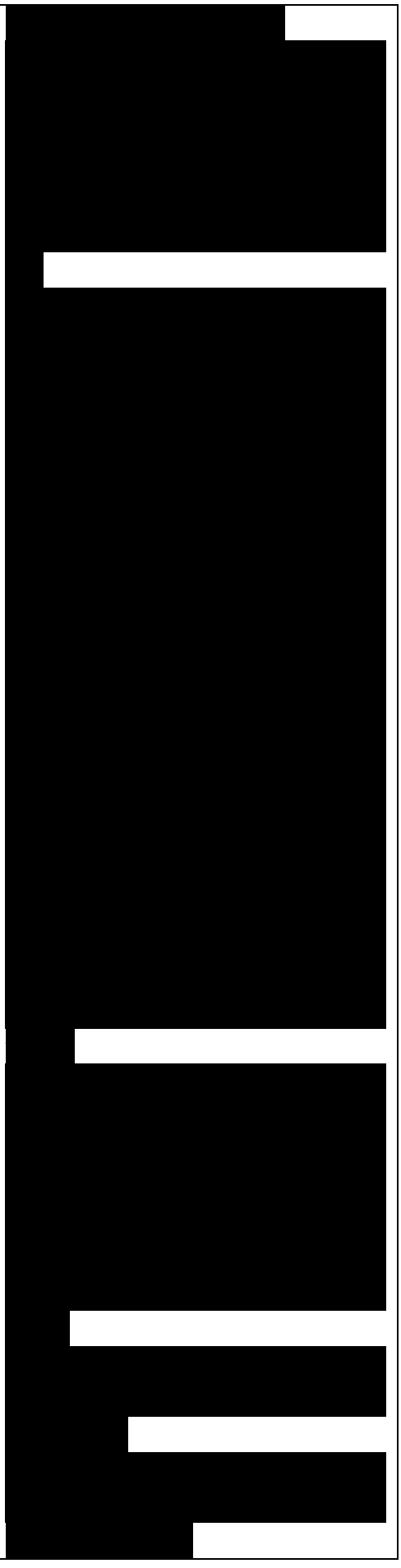


Now we show that the restriction Ka of the perfect model K' of Pa to the bottom stratum Sa = S(Pa) of Pa is the least model of the first layer La = L(Pa) of Pa.

Indeed, otherwise there would exist a non-trivial, minimal component C of Pa such that K'\C is not the least model of the program Pa restricted to C. This follows from the fact that, by Proposition 3.3, non-trivial components C in the bottom stratum of Pa are completely independent of the remaining components in the bottom stratum and from the fact that Pa restricted to the union U of all its trivial minimal components is a positive logic program and thus has the least model. Consequently, since K' is a perfect model, its restriction to U must coincide with the least model.

Therefore there must exist a model T of La restricted to C and an atom A in C such that A £ Ka — T. Let T* be the interpretation of Pa obtained by taking the model K' and modifying it by:

- Making all atoms B in Hp which are greater than A true in T*;
- Replacing the restriction K'\C of K' to the component C by the model T.



It is easy to verify that the fact that the component C is not trivial implies that T* is a model of Pa, which is preferred to K'. This contradiction completes the proof of the theorem. □

In particular, if the instantiated program P is finite then the existence of its perfect model guarantees the existence of its weakly perfect model.

Corollary 4.4 Suppose that a function-free logic program P has a perfect model. Then P also has a weakly perfect model and the two models coincide.

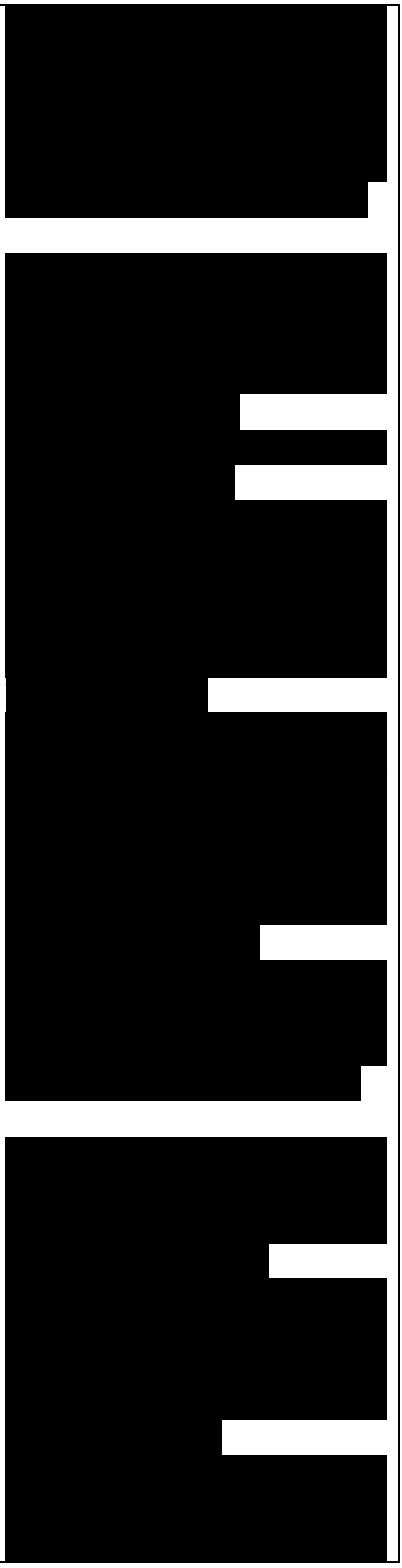
Proof: The instantiation of a non-empty function-free program is always finite and therefore it results in a finite set of components, which therefore contain minimal elements. □

Theorem 4.3 also implies that the weakly perfect model semantics is fully compatible with the perfect model semantics.

Corollary 4.5 If a logic program has both a perfect model and a weakly perfect model then they coincide.

Proof: If a weakly perfect model exists then, by definition, all of the bottom strata Sa, for a < 5(P), are non-empty. □

It can be easily seen that the weakly perfect model semantics is based on the



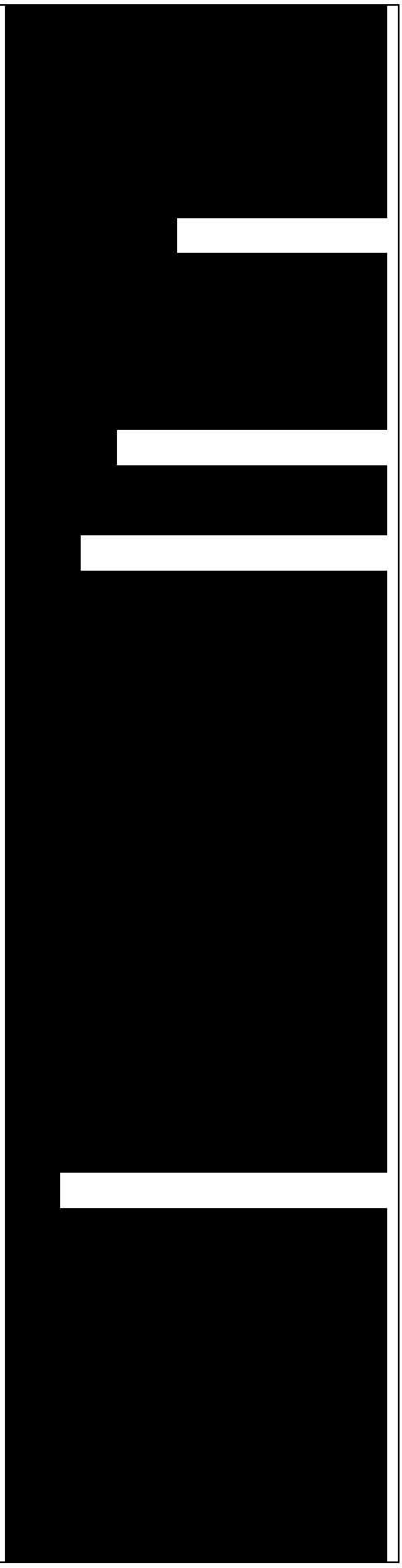
principle of iterated (2-valued) least model semantics. In particular, for programs with a single stratum the weakly perfect model semantics coincides with the least model semantics.

Corollary 4.6 If a logic program P consists of a single stratum (in particular, if it consists of a single component), then for a model M of P:

M is weakly perfect = M is perfect = M is the least model of P.

Proof: Suppose that P consists of a single stratum. By definition, M is the weakly perfect model of P iff M is the least model of P. Moreover, it is obvious from definition of perfect models that the least model of any logic program is always perfect. Therefore, it suffices to show that if M is a perfect model of P then it is the least model of P. This result can be deduced from Theorem 4.3, but the following direct proof is much simpler.

First notice that if P consists of a single non-trivial component C then M is the least model of P. This follows immediately from the definition of perfect models [Prz89b] and the fact that A < B, for any two elements of C. Secondly, if P contains only trivial components then M is



also the least model of P. This follows from the fact that then P is a positive logic program and perfect models of positive logic programs are always least models [Prz89b].

We know that P consists of a single stratum. It follows from Proposition 3.3 that any non-trivial component C of Gp is completely independent of the re¬maining components. Consequently, M\C (M restricted to C) must be a perfect model of P\C (P restricted to C) and thus, in view of the above argument, it must be the least model of P\C. Similarly, the union C of all trivial components of P is completely independent of the remaining components. Consequently, M\C must be a perfect model of P\C and thus the least model of P\C. This implies that M must be the least model of P. □

It was shown in [Prz88a] that perfect models of locally stratified logic pro¬grams are in fact models of McCarthy's prioritized circumscription (a similar result for pointwise circumscription has been proven in [Lif88]). This result can now be stated in greater generality.

Theorem 4.7 If Mp is the

weakly perfect model of P and if {Sa : 0 < a < S(P)} is the corresponding set of strata, then Mp is the unique Herbrand model of prioritized circumscription CIRC(P;Si > S2 >■■■)■

Proof: By definition, models of prioritized circumscription are constructed as follows. We first chose a minimal model M\ of the program P restricted to the first stratum Si, i.e. a minimal model of L\ and we fix it, which is equivalent to applying the Davis-Putnam rule to P, resulting in a reduced program P2. We then choose a minimal program M2 of the program P2 restricted to the second stratum, i.e. a minimal program of L2, etc.

The construction of the weakly perfect model follows exactly the same pro-cedure, but on each level we assume the existence of the least model and thus we have only one choice of a minimal model. This means that there is only one model of prioritized circumscription and that it must coincide with the weakly perfect model. □

In [Prz89b] T.Przymusinski defined the SLS-resolution - a modification of SLDNF-resolution not requiring finite failure - and showed that for stratified programs SLS-resolution is sound and complete with respect to the

perfect model semantics. The above result can be now extended onto the class of weakly stratified programs with the weakly perfect model semantics. We assume that all queries are not floundered (see [Prz89b]) and that all, not necessarily Her¬brand, weakly perfect models of a program are considered.

Theorem 4.8 For weakly stratified programs SLS-resolution is sound and com¬plete with respect to the weakly perfect model semantics.

Proof: The proof is completely analogous to the proof of completeness of SLS-resolution for the perfect model semantics, given in [Prz89b]. □

5 Relation to Other Proposed Semantics

As we mentioned in the introduction, the class of programs with weakly perfect models is different from the class of programs with well-founded semantics - proposed by Van Gelder, Ross and Schlipf in [VGRS90] - and from the class of programs with unique stable model semantics - defined by Gelfond and Lifschitz in [GL88].

Example 5.1 The program P from Example 3.4 has a weakly perfect model, but it does not have either a 2-

valued well-founded model or a unique stable model. □

However, the three approaches are closely related. Van Gelder, Ross and Schlipf [VGRS90] showed that every 2-valued well-founded model is also a unique stable model. We prove the following result:

Theorem 5.1 For weakly stratified programs P, all three semantics coincide, i.e. the weakly perfect model Mp of P is also well-founded and unique stable.

Proof: It suffices to show that Mp is a 2-valued well-founded model of P. We use terminology and notation from [VGRS90]. We will show by induction that for every a < S(P) and for every literal L, if L G Ma then L £ I*, where I* is the limit of sets Ia, defined in [VGRS90].

Suppose that our claim has been proven for all a < 7. We will prove it for 7. By definition, M7 is the least model of L1 and L1 is the bottom level of the program P7 = , ■ p . Let L G M7. If L is positive, then since L1 is a

a <-y 11

positive logic program, L must belong to TLI (0) and therefore, by the inductive assumption, L must belong to TP(I*), and thus to Vp(I*). Since I* is a fixpoint of the operator V, this implies that L G I*. If L = ->A is negative then we define K = {B : -<B G M7}. We will show that K is

unfounded with respect to I*. Since A G K, this will imply that L = ->A G Up (I*), and thus L G Vp(I*) C I*.

Since L1 is positive, if B G K then every clause in L7, with B as its head, must contain a premise from K. Some of the clauses from P, however, may not belong to L7, because they were removed before. But a clause is removed only if it contains as a premise a literal, which is false in the union of the previously constructed models Ma. By the inductive assumption, such premises are also false in I*, which implies that K is unfounded with respect to I* and completes the proof. □

There also exist programs with 2-valued well-founded models which do not have weakly perfect models and programs with unique stable models, which do not have either 2-valued well-founded models or weakly perfect models.

Example 5.2 [[VGRS90]] Let P be given by:

This program has a unique stable model M = {p, 6}, but it does not have either weakly perfect or 2-valued well-founded models. □

Example 5.3 Let P be as follows:

p <r- ^s,q,^r

q i— r,^p r i— p,^q s i— ->p, ->q, ->r.

This program has a 2-valued well-founded model M = {s}, which is automat¬ically

unique stable [VGRS90]. Since it has only one component and does not have the least model, according to Theorem 4.6 it does not have a weakly perfect model. □

For stratified programs, the perfect model semantics, has been shown (see [Prz88b]) to be equivalent to suitable forms of all four major formalizations of non-monotonic reasoning in AI - McCarthy's circumscription [McC80, McC86], Reiter's closed world assumption CWA [Rei78], Moore's autoepistemic logic [Moo85] and Reiter's default theory [Rei80] - thus establishing a close link between the areas of logic programming and non-monotonic reasoning and de¬scribing a relatively large class of theories for which natural forms of different non-monotonic formalisms coincide.

Originally, it seemed that no extension of this result will be possible for classes of logic programs significantly broader than the class of stratified logic programs. The reason appeared to be the fact that, as we have seen, all three proposed extensions of the perfect model semantics - the stable model seman¬tics (based on autoepistemic logic or default logic), the weakly perfect model semantics (based on circumscription or on CWA) and the 3-valued

| | |
|---|---|
| well-founded semantics (seemingly not based on any specific non-monotonic formalism, but defined for all logic programs) - lead to different results.<br>Recently, however, the second author proved [Prz89c] that the well-founded model semantics is also equivalent to suitable forms of all four major formaliza¬tions of non-monotonic reasoning. However, in order to achieve this equivalence, 3-valued extensions of non-monotonic formalisms are needed, which is natural in view of the fact that the well-founded semantics is, in general, 3-valued.<br>Using the concept of dynamic stratification introduced in this paper, the second author also proved [Prz89a] that the (3-valued) well-founded semantics has properties entirely analogous to the properties of the perfect model seman¬tics, which lead to a natural notion of dynamic stratification of an arbitrary logic program. Further, the concept of dynamic stratification has been used in [Prz89a] to extend the definition of SLS-resolution [Prz89b] to the class of all logic programs and to prove that SLS-resolution is sound and complete (for non-floundering queries) with respect to the (3-valued) well-founded model se¬mantics | |

| | |
|---|---|
| (see also [Ros89]).<br>As a result of these developments, a fairly clear picture emerges, showing the existence of two different semantics of logic programs (stable and weakly perfect), corresponding to standard 2-valued non-monotonie formalisms (au¬toepistemic logic or default theory and circumscription or CWA), and a unique semantics (3-valued well-founded), corresponding to 3-valued formalizations of non-monotonic reasoning. The latter semantics is defined for all logic programs, whereas the former two are restricted to their own, more narrow domains. All three semantics, however, coincide in the class of weakly stratified programs.<br>Acknowledgements. The authors are grateful to Michael Gelfond, Vladimir Lifschitz and Allen Van Gelder for suggesting the problem and for helpful discussions. | |