Splitting an operator: Algebraic modularity results for logics with fixpoint semantics

— **Source link** ↗

Joost Vennekens, David Gilis, Marc Denecker

Institutions: Katholieke Universiteit Leuven

Topics: Modal μ-calculus, Higher-order logic, Classical logic, Multimodal logic and Intermediate logic

Related papers:

- Ultimate approximation and its application in nonmonotonic knowledge representation systems

- Fixpoint Semantics for Logic Programming A Survey

- Approximations, stable operators, well-founded fixpoints and applications in nonmonotonic reasoning

- The well-founded semantics for general logic programs

- Uniform semantic treatment of default and autoepistemic logics

Splitting an operator:
Algebraic modularity results for logics with fixpoint semantics

JOOST VENNEKENS, DAVID GILIS and MARC DENECKER

K.U. Leuven

It is well known that, under certain conditions, it is possible to *split* logic programs under stable model semantics, i.e. to divide such a program into a number of different "levels", such that the models of the entire program can be constructed by incrementally constructing models for each level. Similar results exist for other non-monotonic formalisms, such as auto-epistemic logic and default logic. In this work, we present a general, algebraic splitting theory for logics with a fixpoint semantics. Together with the framework of *approximation theory*, a general fixpoint theory for arbitrary operators, this gives us a uniform and powerful way of deriving splitting results for each logic with a fixpoint semantics. We demonstrate the usefulness of these results, by generalizing existing results for logic programming, auto-epistemic logic and default logic.

---

1. INTRODUCTION

An important aspect of human reasoning is that it is often incremental in nature. When dealing with a complex domain, we tend to initially restrict ourselves to a small subset of all relevant concepts. Once these "basic" concepts have been figured out, we then build another, more "advanced", layer of concepts on this knowledge. A quite illustrative example of this can be found in most textbooks on computer networking. These typically present a seven-layered model of the way in which computers communicate. First, in the so-called physical layer, there is only talk of hardware and concepts such as wires, cables and electronic pulses. Once these low-level issues have been dealt with, the resulting knowledge becomes a *fixed* base,

---

upon which a new layer, the data-link layer, is built. This no longer considers wires and cables and so on, but rather talks about packages of information travelling from one computer to another. Once again, after the workings of this layer have been figured out, this information is "taken for granted" and becomes part of the foundation upon which a new layer is built. This process continues all the way up to a seventh layer, the application layer, and together all of these layers describe the operation of the entire system.

In this paper, we investigate a formal equivalent of this method. More specifically, we address the question of whether a formal theory in some non-monotonic language can be *split* into a number of different levels or *strata*, such that the formal semantics of the entire theory can be constructed by succesively constructing the semantics of the various strata. (We use the terms "stratification" and "splitting" interchangeably to denote a division into a number of different levels. This is a more general use of both these terms, than in literature such as Apt et al. [1988] and Gelfond [1987].) Such stratifications are interesting from both a theoretical, knowledge representational and a more practical point of view.

On the more theoretical side, stratification results provide crucial insight into the formal and informal semantics of a language, and hence in its use for knowledge representation. Indeed, the human brain seems unsuited for holding large chunks of unstructured information. When the complexity of a domain increases, we rely on our ability to understand and describe parts of the domain and construct a description of the whole domain by composing the descriptions of its components. Large theories which cannot be understood as somehow being a composition of components, simply cannot be understood by humans. Stratification results are, therefore, important, especially in the context of nonmonotonic languages in which adding a new expression to a theory might affects the meaning of what was already represented. Our results will present cases where adding a new expression is guaranteed *not* to alter the meaning of existing theories.

On the more practical side, computing models of a theory by incrementally constructing models of each of its levels, might offer considerable computational gain. Indeed, suppose that, normally, it takes $t(n)$ time to construct the model(s) of a theory of size n. If we were able to split such a theory into, say, m smaller theories of equal size n/m, we could use this stratification to compute the model(s) of the theory in $m \cdot t(n/m)$ time. As model generation is typically quite hard, i.e. $t(n)$ is a large function of n, this could provide quite a substantial improvement. Of course, much depends of the value of m. Indeed, in the worst case, the theory would allow only the trivial stratification in which the entire theory is a single level, i.e. $m = 1$, which obviously does not lead to any gain. However, because (as argued above) human knowlegde tends to exhibit a more modular structure, we would expect real knowledge bases to be rather well-behaved in this respect.

It is therefore not surprising that stratifiability and related concepts, such as e.g. Dix's notion of "modularity" [Dix 1995], have already been intensively studied. It is therefore not surprising that this issue has already been intensively studied. Indeed, splitting results have been proven for auto-epistemic logic under the semantics of expansions [Gelfond and Przymusinska 1992; Niemelä and Rintanen 1994] default logic under the semantics of extensions [Turner 1996] and various kinds of logic

programs under the stable model semantics [Lifschitz and Turner 1994; Erdoğan and Lifschitz 2004; Eiter et al. 1997]. In all of these works, stratification is seen as a syntactical property of a theory in a certain language under a certain formal semantics.

In this work, we take a different approach to studying this topic. The semantics of several (non-monotonic) logics can be expressed through fixpoint characterizations in some lattice of semantic structures. We will study the stratification of these semantical operators themselves. As such, we are able to develop a general theory of stratification at an abstract, algebraic level and apply its results to each formalism which has a fixpoint semantics.

This approach is especially powerful when combined with the framework of *approximation theory*, a general fixpoint theory for arbitrary operators, which has already proved highly useful in the study of non-monotonic reasoning. It naturally captures, for instance, (most of) the common semantics of logic programming [Denecker et al. 2000], auto-epistemic logic [Denecker et al. 2003] and default logic [Denecker et al. 2003]. As such, studying stratification within this framework, allows our abstract results to be directly and easily applicable to logic programming, auto-epistemic logic and default logic.

To make this a bit more concrete, we will now briefly sketch our method of derving splitting results. Approximation theory defines a family of different kinds of fixpoints of operators and shows that, using a suitable class of operators, these fixpoints correspond to a family of semantics for a number of different logics. We will introduce the concept of a *stratifiable operator* and prove that such operators can be split into a number of smaller *component operators*, in such a way that the different kinds of fixpoints of the original operator can be constructed by incrementally constructing the corresponding fixpoints of its component operators. These algebraic results will then be used to derive concrete splitting results for logic programming, auto-epistemic logic and default logic. To do this, we will follow these two steps:

— Firstly, we have to determine *syntactical* conditions which suffice to ensure that every operator corresponding to a theory which satisfies these conditions, is in fact a stratifiable operator. This tells us that the models of such a theory under various semantics, i.e. the various kinds of fixpoints of the associated operator, can be constructed by incrementally constructing the corresponding fixpoints of the components of this operator.

— Secondly, we also need to provide a precise, computable characterization of the components of stratifiable operators. This will be done by presenting a *syntactical* method of deriving a number of smaller theories from the original theory and showing that the components of the original operator are precisely the operators associated with these new theories.

So, in other words, using the algebraic characterization of the semantics of a number of different logics by approximation theory, our algebraic results show how splitting can be done on a semantical level, and deriving concrete splitting results for a specific logic simply boils down to determining which syntactical notions correspond to our semantical splitting concepts.

Studying stratification at this more *semantical* level has three distinct advantages. First of all, it avoids duplication of effort, as the same algebraic theory takes

care of stratification in logic programming, auto-epistemic logic, default logic and indeed any logic with a fixpoint semantics. Secondly, our results can be used to easily extend existing results to other (fixpoint) semantics of the aforementioned languages. Finally, our work also offers greater insight into the general principles underlying various known stratification results, as we are able to study this issue in itself, free of being restricted to a particular syntax or semantics.

This paper is structured in the following way. In Section 2, some basic notions from lattice theory are introduced and a brief introduction to the main concepts of approximation theory is given. Section 3 is the main part of this work, in which we present our algebraic theory of stratifiable operators. In Section 4, we then show how these abstract results can be applied to logic programming, auto-epistemic logic and default logic, thus proving generalizations of existing results. Parts of this paper have been published as Vennekens et al. [2004b], which contains the results concerning logic programming, and Vennekens et al. [2004a], which contains a shorter version of the results for auto-epistemic logic.

2. PRELIMINARIES

In this section, we introduce some basic concepts which will be needed later on. Section 2.1 summarizes a number of well known definitions and results from lattice theory, while Section 2.2 contains an overview of the framework of *approximation theory*.

2.1 Orders and lattices

A binary relation $\leq$ on a set S is a *partial order* if it is reflexive, transitive and anti-symmetric. An element $x \in S$ is a *central element* if it is comparable to each other element of S, i.e. $x \leq y$ or $x \geq y$ for each $y \in S$. For each subset R of S, an element l of S, such that $l \leq r$ for all $r \in R$ is a *lower bound* of R. An element g in S such that g is a lower bound of R and for each other lower bound l of R, $l \leq g$, is called the *greatest lower bound*, denoted $glb(R)$, of R. Similarly, an element u such that for each $r \in R$, $u \geq r$ is an *upper bound* of R and if one such upper bound is less or equal to each other upper bound of R, it is the *least upper bound $lub(R)$* of R.

A partial order $\leq$ on a set S is *well-founded* if each non-empty subset R of S has a minimal element; it is *total* if every two elements $x, y \in S$ are comparable, i.e. $x \leq y$ or $x \geq y$.

A pair $\langle L, \leq \rangle$ is a *lattice* if $\leq$ is a partial order on a non-empty set L, such that each two elements x, y of L have a greatest lower bound $glb(x, y)$ and a least upper bound $lub(x, y)$. A lattice $\langle L, \leq \rangle$ is *complete* if each subset L' of L has a greatest lower bound $glb(L')$ and least upper bound $lub(L')$. By definition, a complete lattice has a minimal (or *bottom*) element $\perp$ and a maximal (or *top*) element $\top$. Often, we will not explicitly mention the partial order $\leq$ of a lattice $\langle L, \leq \rangle$ and simply speak of the lattice L.

An *operator* is a function from a lattice to itself. An operator O on a lattice L is monotone if for each $x, y \in L$, such that $x \leq y$, $O(x) \leq O(y)$. An element x in L is a *fixpoint* of O if $O(x) = x$. We denote the set of all fixpoint of O by $fp(O)$. A fixpoint x of L, such that for each other fixpoint y of L, $x \leq y$, is *the least fixpoint $lfp(O)$* of O. It can be shown [Tarski 1955] that each monotone operator on a

complete lattice has a unique least fixpoint.

2.2 Approximation theory

Approximation theory is a general fixpoint theory for arbitrary operators, which generalizes ideas found in, among others, Baral and Subrahmanian [1991], Ginsberg [1988] and Fitting [1991]. Our presentation of this theory is based on Denecker et al. [2000]. However, we will introduce a slightly more general definition of approximation. For a comparison between approximation theory and related approaches, we refer to Denecker et al. [2000] and Denecker et al. [2003].

Let $\langle L, \leq \rangle$ be a lattice. An element (x, y) of the square L^2 of the domain of such a lattice, can be seen as denoting an interval $[x, y] = \{z \in L \mid x \leq z \leq y\}$. Using this intuition, we can derive a *precision* order $\leq_p$ on the set L^2 from the order $\leq$ on L: for each $x, y, x', y' \in L, (x, y) \leq_p (x', y')$ iff $x \leq x'$ and $y' \leq y$. Indeed, if $(x, y) \leq_p (x', y')$, then $[x, y] \supseteq [x', y']$. It can easily be shown that $\langle L^2, \leq_p \rangle$ is also a lattice, which we will call the *bilattice* corresponding to L. Moreover, if L is complete, then so is L^2. As an interval $[x, x]$ contains precisely one element, namely x itself, elements (x, x) of L^2 are called *exact*. The set of all exact elements of L^2 forms a natural embedding of L in L^2. A pair (x, y) only corresponds to a non-empty interval if $x \leq y$. Such pairs are called *consistent*.

Approximation theory is based on the study of operators on bilattices L^2 which are monotone w.r.t. the precision order $\leq_p$. Such operators are called *approximations*. For an approximation A and elements x, y of L, we denote by $A^1(x, y)$ and $A^2(x, y)$ the unique elements of L, for which $A(x, y) = (A^1(x, y), A^2(x, y))$. An approximation *approximates* an operator O on L if for each $x \in L$, $A(x, x)$ contains $O(x)$, i.e. $A^1(x, x) \leq O(x) \leq A^2(x, x)$. An *exact* approximation is one which maps exact elements to exact elements, i.e. $A^1(x, x) = A^2(x, x)$ for all $x \in L$. Similarly, a *consistent* approximation maps consistent elements to consistent elements, i.e. if $x \leq y$ then $A^1(x, y) \leq A^2(x, y)$. If an approximation is not consistent, it cannot approximate any operator. Each exact approximation is also consistent and approximates a unique operator O on L, namely that which maps each $x \in L$ to $A^1(x, x)$. An approximation is *symmetric* if for each pair $(x, y) \in L^2$, if $A(x, y) = (x', y')$ then $A(y, x) = (y', x')$. Each symmetric approximation is also exact.

For an approximation A on L^2, the following two operators on L can be defined: the function $A^1(\cdot, y)$ maps an element $x \in L$ to $A^1(x, y)$, i.e. $A^1(\cdot, y) = \lambda x.A^1(x, y)$, and the function $A^2(x, \cdot)$ maps an element $y \in L$ to $A^2(x, y)$, i.e. $A^2(x, \cdot) = \lambda y.A^2(x, y)$. As all such operators are monotone, they all have a unique least fixpoint. We define an operator $C_A^\downarrow$ on L, which maps each $y \in L$ to $lfp(A^1(\cdot, y))$ and, similarly, an operator $C_A^\uparrow$, which maps each $x \in L$ to $lfp(A^2(x, \cdot))$. $C_A^\downarrow$ is called the *lower stable operator* of A, while $C_A^\uparrow$ is the *upper stable operator* of A. Both these operators are anti-monotone. Combining these two operators, the operator $\mathcal{C}_A$ on L^2 maps each pair (x, y) to $(C_A^\downarrow(y), C_A^\uparrow(x))$. This operator is called the *partial stable operator* of A. Because the lower and upper partial stable operators $C_A^\downarrow$ and $C_A^\uparrow$ are anti-monotone, the partial stable operator $\mathcal{C}_A$ is monotone. Note that if an approximation A is symmetric, its lower and upper partial stable operators will always be equal, i.e. $C_A^\downarrow = C_A^\uparrow$.

An approximation A defines a number of different fixpoints: the least fixpoint

of an approximation A is called its *Kripke-Kleene fixpoint*, fixpoints of its partial stable operator $\mathcal{C}_A$ are *stable fixpoints* and the least fixpoint of $\mathcal{C}_A$ is called the *well-founded fixpoint* of A. As shown in Denecker et al. [2000] and Denecker et al. [2003], these fixpoints correspond to various semantics of logic programming, auto-epistemic logic and default logic.

Finally, it should be noted that the concept of an approximation as defined in Denecker et al. [2000] corresponds to our definition of a *symmetric* approximation.

3. STRATIFICATION OF OPERATORS

In this section, we develop a theory of stratifiable operators. We will, in section 3.2, investigate operators on a special kind of lattice, namely *product lattices*, which will be introduced in section 3.1. In section 3.3, we then return to approximation theory and discuss stratifiable approximations on product lattices.

3.1 Product lattices

We begin by defining the notion of a *product set*, which is a generalization of the well-known concept of cartesian products.

Definition 3.1. Let I be a set, which we will call the *index set* of the product set, and for each $i \in I$, let S_i be a set. The *product set* $\bigotimes_{i \in I} S_i$ is the following set of functions:

$$\bigotimes_{i \in I} S_i = \{ f \mid f : I \to \bigcup_{i \in I} S_i \text{ such that } \forall i \in I : f(i) \in S_i \}.$$

Intuitively, a product set $\bigotimes_{i \in I} S_i$ contains all ways of selecting one element from each set S_i. As such, if the index set I is a set with n elements, e.g. the set $\{1, \dots, n\}$, the product set $\bigotimes_{i \in I} S_i$ is simply (isomorphic to) the cartesian product $S_1 \times \cdots \times S_n$.

Definition 3.2. Let I be a set and for each $i \in I$, let $\langle S_i, \leq_i \rangle$ be a partially ordered set. The *product order* $\leq_\otimes$ on the set $\otimes_{i \in I} S_i$ is defined by $\forall x, y \in \bigotimes_{i \in I} S_i$:

$$x \leq_\otimes y \qquad iff \qquad \forall i \in I : x(i) \leq_i y(i).$$

It can easily be shown that if all of the partially ordered sets S_i are (complete) lattices, the product set $\otimes_{i \in I} S_i$, together with its product order $\leq_\otimes$, is also a (complete) lattice. We therefore refer to the pair $\langle \otimes_{i \in I} S_i, \leq_\otimes \rangle$ as the *product lattice* of lattices S_i.

From now on, we will only consider product lattices with a *well-founded* index set, i.e. index sets I with a partial order $\preceq$ such that each non-empty subset of I has a $\preceq$-minimal element. This will allow us to use inductive arguments in dealing with elements of product lattices. Most of our results, however, also hold for index sets with an arbitrary partial order; if a certain proof depends on the well-foundedness of I, we will always explicitly mention this.

In the next sections, the following notations will be used. For a function $f : A \to B$ and a subset A' of A, we denote by $f|_{A'}$ the restriction of f to A', i.e. $f|_{A'} : A' \to B : a' \mapsto f(a')$. For an element x of a product lattice $\otimes_{i \in I} L_i$ and an $i \in I$, we abbreviate $x|_{\{j \in I \mid j \preceq i\}}$ by $x|_{\preceq i}$. We also use similar abbreviations $x|_{\prec i}$, $x|_i$ and $x|_{\npreceq i}$. If i is a minimal element of the well-founded set I, $x|_{\prec i}$ is defined as the empty

function. For each index i, the set $\{x|_{\preceq i} \mid x \in L\}$, ordered by the appropriate restriction $\leq_{\otimes}|_{\preceq i}$ of the product order, is also a lattice. Clearly, this sublattice of L is isomorphic to the product lattice $\otimes_{j \preceq i} L_i$. We denote this sublattice by $L|_{\preceq i}$ and use a similar notation $L|_{\prec i}$ for $\otimes_{j \prec i} L_i$.

If f, g are functions $f : A \to B$, $g : C \to D$ and the domains A and C are disjoint, we denote by $f \sqcup g$ the function from $A \cup C$ to $B \cup D$, such that for all $a \in A$, $(f \sqcup g)(a) = f(a)$ and for all $c \in C$, $(f \sqcup g)(c) = g(c)$. Furthermore, for any g whose domain is disjoint from the domain of f, we call $f \sqcup g$ an *extension* of f. For each element x of a product lattice L and each index $i \in I$, the extension $x|_{\prec i} \sqcup x|_i$ of $x|_{\prec i}$ is clearly equal to $x|_{\preceq i}$. For ease of notation, we sometimes simply write $x(i)$ instead of $x|_i$ in such expressions, i.e. we identify an element a of the ith lattice L_i with the function from $\{i\}$ to L_i which maps i to a. Similarly, $x|_{\prec i} \sqcup x(i) \sqcup x|_{\npreceq i} = x$.

We will use the symbols x, y to denote elements of an entire product lattice L; a, b to denote elements of a single level L_i and u, v to denote elements of $L|_{\prec i}$.

3.2 Operators on product lattices

Let $\langle I, \preceq \rangle$ be a well-founded index set and let $L = \otimes_{i \in I} L_i$ be the product lattice of lattices $\langle L_i, \leq_i \rangle_{i \in I}$. Intuitively, an operator O on L is stratifiable over the order $\preceq$, if the value $(O(x))(i)$ of $O(x)$ in the ith stratum only depends on values $x(j)$ for which $j \preceq i$. This is formalized in the following definition.

Definition 3.3. An operator O on a product lattice L is *stratifiable* iff $\forall x, y \in L, \forall i \in I$: if $x|_{\preceq i} = y|_{\preceq i}$ then $O(x)|_{\preceq i} = O(y)|_{\preceq i}$.

It is also possible to characterize stratifiablity in a more constructive manner. The following theorem shows that stratifiablity of an operator O on a product lattice L is equivalent to the existence of a family of operators on each lattice L_i (one for each partial element u of $L|_{\prec i}$), which mimics the behaviour of O on this lattice.

PROPOSITION 3.4. *Let O be an operator on a product lattice L. O is stratifiable iff for each $i \in I$ and $u \in L|_{\prec i}$ there exists a unique operator O_i^u on L_i, such that for all $x \in L$:*

$$\text{If } x|_{\prec i} = u \text{ then } (O(x))(i) = O_i^u(x(i)).$$

PROOF. To prove the implication from left to right, let O be a stratifiable operator, $i \in I$ and $u \in L|_{\prec i}$. We define the operator O_i^u on L_i as

$$O_i^u : L_i \to L_i : a \mapsto (O(y))(i),$$

with y some element of L extending $u \sqcup a$. Because of the stratifiability of O, this operator is well-defined and it trivially satisfies the required condition.

To prove the other direction, suppose the right-hand side of the equivalence holds and let x, x' be elements of L, such that $x|_{\preceq i} = x'|_{\preceq i}$. Then for each $j \preceq i$, $(O(x))(j) = O_j^{x|_{\prec j}}(x(j)) = O_j^{x'|_{\prec j}}(x'(j)) = (O(x'))(j)$. □

The operators O_i^u are called the *components* of O. Their existence allows us to already prove one of the main theorems of this paper, which states that is possible to construct the fixpoints of a stratifiable operator in a bottom-up manner w.r.t. the well-founded order $\preceq$ on the index set.

THEOREM 3.5. *Let O be a stratifiable operator on a product lattice L. Then for each $x \in L$:*

$$x \text{ is a fixpoint of } O \qquad iff \qquad \forall i \in I : x(i) \text{ is a fixpoint of } O_i^{x|_{\prec i}}.$$

PROOF. Follows immediately from proposition 3.4. $\square$

If O is a monotone operator on a complete lattice, we are often interested in its *least* fixpoint. This can also be constructed by means of the least fixpoints of the components of O. Such a construction of course requires each component to actually have a least fixpoint as well. We will therefore first show that the components of a monotone operator are also monotone.

PROPOSITION 3.6. *Let O be a stratifiable operator on a product lattice L, which is monotone w.r.t. the product order $\leq_\otimes$. Then for each $i \in I$ and $u \in L|_{\prec i}$, the component $O_i^u : L_i \to L_i$ is monotone w.r.t. to the order $\leq_i$ of the ith lattice L_i of L.*

PROOF. Let i be an index in I, u an element of $L|_{\prec i}$ and a, b elements of L_i, such that $a \leq_i b$. Let $x, y \in L$, such that x extends $u \sqcup a$, y extends $u \sqcup b$ and for each $j \npreceq i$, $x(j) = y(j)$. Because of the definition of $\leq_\otimes$, clearly $x \leq_\otimes y$ and therefore $\forall j \in I : O_j^{x|_{\prec j}}(x(j)) = (O(x))(j) \leq_j (O(y))(j) = O_j^{y|_{\prec j}}(y(j))$, which, taking $j = i$, implies $O_i^u(a) \leq_i O_i^u(b)$. $\square$

Now, we can prove that the least fixpoints of the components of a monotone stratifiable operator indeed form the least fixpoint of the operator itself. We will do this, by first proving the following, slightly more general theorem, which we will be able to reuse later on.

PROPOSITION 3.7. *Let O be a monotone operator on a complete product lattice L and let for each $i \in I$, $u \in L|_{\prec i}$, P_i^u be a monotone operator on L_i (not necessarily a component of O), such that:*

$$x \text{ is a fixpoint of } O \qquad iff \qquad \forall i \in I : x(i) \text{ is a fixpoint of } P_i^{x|_{\prec i}}.$$

Then the following equivalence also holds:

$x \text{ is the least fixpoint of } O \qquad iff \qquad \forall i \in I : x(i) \text{ is the least fixpoint of } P_i^{x|_{\prec i}}.$

PROOF. To prove the implication from left to right, let x be the least fixpoint of O and let i be an arbitrary index in I. We will show that for each fixpoint a of $P_i^{x|_{\prec i}}$, $a \geq x(i)$. So, let a be such a fixpoint. We can inductively extend $x|_{\prec i} \sqcup a$ to an element y of L by defining for all $j \npreceq i$, $y(j)$ as $lfp(P_j^{y|_{\prec j}})$. Because of the well-foundedness of $\preceq$, y is well defined. Furthermore, y is clearly also a fixpoint of O. Therefore $x \leq y$ and, by definition of the product order on L, $x(i) \leq_i y(i) = a$.

To prove the other direction, let x be an element of L, such that, for each $i \in I$, $x(i)$ is the least fixpoint of $P_i^{x|_{\prec i}}$. Now, let y be the least fixpoint of O. To prove that $x = y$, it suffices to show that for each $i \in I$, $x|_{\preceq i} = y|_{\preceq i}$. We will prove this by by induction on the well-founded order $\preceq$ of I. If i is a minimal element of I, the proposition trivially holds. Now, let i be an index which is not the minimal element of I and assume that for each $j \prec i$, $x|_{\preceq j} = y|_{\preceq j}$. It suffices to show

that $x(i) = y(i)$. Because y is a fixpoint of O, $y(i)$ is fixpoint of $P_i^{y|_{\prec i}}$. As the induction hypothesis implies that $x|_{\prec i} = y|_{\prec i}$, $y(i)$ is a also fixpoint of $P_i^{x|_{\prec i}}$ and therefore $x(i) \leq y(i)$. However, because x is also a fixpoint of O and therefore must be greater than the least fixpoint y of O, the definition of the product order on L implies that $x(i) \geq y(i)$ as well. Therefore $x(i) = y(i)$. $\square$

It is worth noting that the condition that the order $\preceq$ on I should be well-founded is necessary for this proposition to hold. Indeed, consider for example the product lattice $L = \bigotimes_{z \in \mathbb{Z}} \{0,1\}$, with $\mathbb{Z}$ the integers ordered by their usual, non-well-founded order. Let O be the operator mapping each $x \in L$ to the element $y : \mathbb{Z} \to \{0,1\}$ of L, which maps each $z \in \mathbb{Z}$ to 0 if $x(z-1) = 0$ and to 1 otherwize. This operator is stratifiable over the order $\leq$ of $\mathbb{Z}$ and its components are the family of operators O_z^u, with $z \in \mathbb{Z}$ and $u \in L|_{\leq z}$, which are defined as mapping both 0 and 1 to 0 if $u(z-1) = 0$ and to 1 otherwize. Clearly, the bottom element $\bot_L$ of L, which maps each $z \in \mathbb{Z}$ to 0, is the least fixpoint of O. However, the element $x \in L$ which maps each $z \in \mathbb{Z}$ to 1 satisfies the condition that for each $z \in \mathbb{Z}$, $x(z)$ is the least fixpoint of $P_z^{x|_{<z}}$, but is clearly not the least fixpoint of O.

Together with theorem 3.5 and proposition 3.6, this proposition of course implies that for each stratifiable operator O on a product lattice L, an element $x \in L$ is the least fixpoint of O iff $\forall i \in I$, $x(i)$ is the least fixpoint of $O_i^{x|_{\prec i}}$. In other words, the least fixpoint of a stratifiable operator can also be incrementally constructed.

3.3 Approximations on product lattices

In section 2.2, we introduced several concepts from approximation theory, pointing out that we are mainly interested in studying Kripke-Kleene, stable and well-founded fixpoints of approximations. Similar to our treatment of general operators in the previous section, we will in this section investigate the relation between these various fixpoints of an approximation and its components. In doing so, it will be convenient to switch to an alternative representation of the bilattice L^2 of a product lattice $L = \otimes_{i \in I} L_i$. Indeed, this bilattice is clearly isomorphic to the structure $\otimes_{i \in I} L_i^2$, i.e. to a product lattice of bilattices. From now on, we will not distinguish between these two representations. More specifically, when viewing A as a stratifiable operator, it will be convenient to consider its domain equal to $\otimes_{i \in I} L_i^2$, while when viewing A as an approximation, the representation $(\otimes_{i \in I} L_i)^2$ is more natural.

Obviously, this isomorphism and the results of the previous section already provide a way of constructing the Kripke-Kleene fixpoint of a stratifiable approximation A, by means of its components A_i^u. Also, it is clear that if A is both exact and stratifiable, the unique operator O approximated by A is stratifiable as well. Indeed, this is a trivial consequence of the fact that $A(x,x) = (O(x), O(x))$ for each $x \in L$.

These results leave only the stable and well-founded fixpoints of A to be investigated. We will first examine the operators $A^1(\cdot, y)$ and $A^2(x, \cdot)$, and then move on to the lower and upper stable operators $C_A^{\downarrow}$ and $C_A^{\uparrow}$, before finally getting to the partial stable operator $\mathcal{C}_A$ itself.

PROPOSITION 3.8. *Let L be a product lattice and let $A : L^2 \to L^2$ be a stratifiable*

approximation. Then, for each $x, y \in L$, the operators $A^1(\cdot, y)$, $A^2(x, \cdot)$ are also stratifiable. Moreover, for each $i \in I$, $u \in L|_{\prec i}$, the components of these operators are:

$$(A^1(\cdot, y))_i^u = (A_i^{(u, y|_{\prec i})})^1(\cdot, y(i));$$
$$(A^2(x, \cdot))_i^u = (A_i^{(x|_{\prec i}, u)})^2(x(i), \cdot).$$

PROOF. Let x, y be elements of L, i an element of I. Then, because A is stratifiable, $(A(x, y))(i) = (A_i^{(x, y)|_{\prec i}})(x(i), y(i))$. From this, the two equalities follow. □

In the previous section, we showed that the components of a monotone operator are monotone as well (proposition 3.6). This result obviously implies that the components A_i^u of a stratifiable approximation are also approximations. Therefore, such a component A_i^u itself has a lower and uppers stable operator $C_{A_i^u}^{\downarrow}$ and $C_{A_i^u}^{\uparrow}$ as well. It turns out that the lower and upper stable operators of the components of A, characterize the components of the lower and upper stable operators of A.

PROPOSITION 3.9. *Let L be a product lattice and let A be a stratifiable approximation on L^2. Then the operators $C_A^{\downarrow}$ and $C_A^{\uparrow}$ are also stratifiable. Moreover, for each $x, y \in L$,*

$$x = C_A^{\downarrow}(y) \qquad \textit{iff} \qquad \textit{for each } i \in I, x(i) = C_{A_i^{(x,y)|_{\prec i}}}^{\downarrow}(y(i));$$

$$y = C_A^{\uparrow}(x) \qquad \textit{iff} \qquad \textit{for each } i \in I, y(i) = C_{A_i^{(x,y)|_{\prec i}}}^{\uparrow}(x(i)).$$

PROOF. Let x, y be elements of L. Because $A^1(\cdot, y)$ is stratifiable (proposition 3.8), the corollary to proposition 3.7 implies that $x = C_A^{\downarrow}(y) = lfp(A^1(\cdot, y))$ iff for each $i \in I$, $x(i) = lfp((A^1(\cdot, y))_i^{x|_{\prec i}})$. Because of proposition 3.8, this is in turn equivalent to for each $i \in I$, $x(i) = lfp((A_i^{(x,y)|_{\prec i}})^1(\cdot, y(i))) = C_{A_i^{(x,y)|_{\prec i}}}^{\downarrow}(y(i))$. The proof of the second equivalence is similar. □

This proposition shows how, for each $x, y \in L$, $C_A^{\downarrow}(y)$ and $C_A^{\uparrow}(x)$ can be be constructed incrementally from the upper and lower stable operators corresponding to the components of A. This result also implies a similar property for the partial stable operator $\mathcal{C}_A$ of an approximation A.

PROPOSITION 3.10. *Let L be a product lattice and let $A : L^2 \to L^2$ be a stratifiable approximation. Then the operator $\mathcal{C}_A$ is also stratifiable. Moreover, for each $x, x', y, y' \in L$, the following equivalence holds:*

$$(x', y') = \mathcal{C}_A(x, y) \qquad iff \qquad \forall i \in I : \begin{cases} x'(i) = C_{A_i^{(x',y)|_{\prec i}}}^{\downarrow}(y(i)); \\ y'(i) = C_{A_i^{(x,y')|_{\prec i}}}^{\uparrow}(x(i)). \end{cases}$$

PROOF. The above equivalence follows immediately from proposition 3.9. To prove the stratifiability of $\mathcal{C}_A$, suppose that $x_1, y_1, x_2, y_2 \in L$, such that $(x_1, y_1)|_{\preceq i} = (x_2, y_2)|_{\preceq i}$. Let $(x_1', y_1') = \mathcal{C}_A(x_1, y_1)$ and $(x_2', y_2') = \mathcal{C}_A(x_2, y_2)$. It suffices to show that $\forall j \preceq i$, $x_1'(j) = x_2'(j)$ and $y_1'(j) = y_2'(j)$. We show this by induction on $\preceq$. Firstly, if j is minimal, then $C_{A_j}^{\downarrow}(y_1(j)) = C_{A_j}^{\downarrow}(y_2(j))$ and $C_{A_j}^{\uparrow}(x_1(j)) = C_{A_j}^{\uparrow}(x_2(j))$.

Secondly, if j is not minimal, then $C^{\downarrow}_{A_j^{(x_1',y_1)|_{\prec j}}}(y_1(j)) = C^{\downarrow}_{A_j^{(x_2',y_2)|_{\prec j}}}(y_2(j))$ and $C^{\uparrow}_{A_j^{(x_1,y_1')|_{\prec j}}}(x_1(j)) = C^{\uparrow}_{A_j^{(x_2,y_2')|_{\prec j}}}(x_2(j))$, because obviously $y_1|_{\preceq j} = y_2|_{\preceq j}$ and $x_1|_{\preceq j} = x_2|_{\preceq j}$, while the induction hypothesis states that $x_1'|_{\prec j} = x_2'|_{\prec j}$ and $y_1'|_{\prec j} = y_2'|_{\prec j}$. $\square$

It should be noted that the components $(\mathcal{C}_A)_i^{(u,v)}$ of the partial stable operator of a stratifiable approximation A are (in general) not equal to the partial stable operators $\mathcal{C}_{A_i^{(u,v)}}$ of the components of A. Indeed, $(\mathcal{C}_A)_i^{(u,v)} = ((C_A^{\downarrow})_i^v, (C_A^{\uparrow})_i^u))$, whereas $\mathcal{C}_{A_i^{(u,v)}} = (C^{\downarrow}_{A_i^{(u,v)}}, C^{\uparrow}_{A_i^{(u,v)}})$. Clearly, these two pairs are, in general, not equal, as $(C_A^{\downarrow})_i^v$ ignores the argument u, which does appear in $C^{\downarrow}_{A_i^{(u,v)}}$. We can, however, characterize the fixpoints of $\mathcal{C}_A$, i.e. the partial stable fixpoints of A, by means of the partial stable fixpoints of the components of A.

THEOREM 3.11. *Let L be a product lattice and let $A : L^2 \to L^2$ be a stratifiable approximation. Then for each element (x,y) of L^2:*

(x,y) *is a fixpoint of* $\mathcal{C}_A$ *iff* $\forall i \in I : (x,y)(i)$ *is a fixpoint of* $\mathcal{C}_{A_i^{(x,y)|_{\prec i}}}$.

PROOF. Let x, y be elements of L, such that $(x,y) = \mathcal{C}_A(x,y)$. By proposition 3.10, this is equivalent to for each $i \in I$, $x = C^{\downarrow}_{A_i^{(x,y)|_{\prec i}}}(y(i))$ and $y = C^{\uparrow}_{A_i^{(x,y)|_{\prec i}}}(x(i))$. $\square$

By proposition 3.7, this theorem has the following corollary:

COROLLARY 3.12. *Let L be a product lattice and let $A : L^2 \to L^2$ be a stratifiable approximation. Then for each element (x,y) of L^2:*

$(x,y) = lfp(\mathcal{C}_A)$ *iff* $\forall i \in I : (x,y)(i) = lfp(\mathcal{C}_{A_i^{(x,y)|_{\prec i}}})$.

Putting all of this together, the main results of this section can be summarized as follows. If A is a stratifiable approximation on a product lattice L, then a pair (x,y) is a fixpoint, Kripke-Kleene fixpoint, stable fixpoint or well-founded fixpoint of A iff for each $i \in I$, $(x(i),y(i))$ is a fixpoint, Kripke-Kleene fixpoint, stable fixpoint or well-founded fixpoint of the component $A_i^{(x,y)|_{\prec i}}$ of A. Moreover, if A is exact then an element $x \in L$ is a fixpoint of the unique operator O approximated by A iff for each $i \in I$, $(x(i),x(i))$ is a fixpoint of the component $A_i^{(x,x)|_{\prec i}}$ of A. These characterizations give us a way of incrementally constructing each of these fixpoints.

4. APPLICATIONS

The general, algebraic framework of stratifiable operators developed in the previous section, allows us to easily and uniformly prove splitting theorems for logics with a fixpoint semantics. We will demonstrate this, by applying the previous results to logic programming (Section 4.1), auto-epistemic logic (Section 4.2) and default logic (Section 4.3).

As noted earlier, for each of these formalisms there exists a class of approximations, such that the various kinds of fixpoints of the approximation A_T associated with a theory T correspond to the models of T under various semantics for this formalism.

In the case of logic programming, these approximations operate on a (lattice isomorphic to) product lattice. Therefore, it is possible to derive splitting results for logic programming by the following method: First, we need to identify syntactical conditions, such that for each logic program P satisfying these conditions, the corresponding approximation A_P is stratifiable. Our results then show that the fixpoints, least fixpoint, stable fixpoints and well-founded fixpoint of such an approximation A_P (which correspond to the models of P under various semantics for logic programming) can be incrementally constructed from the components of A_P. Of course, in order for this result to be of any practical use, one also needs to be able to actually construct these components. Therefore, we will also present a procedure of deriving new programs P' from the original program P, such that the approximations associated with these programs P' are precisely those components.

For auto-epistemic logic and default logic the situation is, however, slightly more complicated, because the approximations which define the semantics of these formalism do not operate on a product lattice. Therefore, in these cases, we cannot simply follow the above procedure. Instead, we need to preform the additional step of first finding approximations $\tilde{A}_T$ on a product lattice "similar" to the original lattice, which "mimic" the original approximations A_T of theories T. The results of this section then show that the various kinds of fixpoints of A_T can be found from the corresponding fixpoints of $\tilde{A}_T$. Therefore, we can split theories T by stratifying these new approximations $\tilde{A}_T$ and, as these $\tilde{A}_T$ *are* operators on a product lattice, this can be done by the above procedure. In other words, we then just need to determine syntactical conditions which suffice to ensure that $\tilde{A}_T$ is stratifiable and present a way of constructing new theories T' from T, such that the components of $\tilde{A}_T$ correspond to approximations associated with these new theories.

4.1   Logic Programming

4.1.1   *Syntax and semantics.* For simplicity, we will only deal with propositional logic programs. Let Σ be an alphabet, i.e. a collection of symbols which are called *atoms*. A *literal* is either an atom p or the negation $\neg q$ of an atom q. A logic program is a set of *clauses* of the following form:

$$h \leftarrow b_1, \ldots, b_n.$$

Here, h is a atom and the b_i are literals. For such a clause r, we denote by $head(r)$ the atom h and by $body(r)$ the set $\{b_1, \ldots, b_n\}$ of literals.

Logic programs can be interpreted in the lattice $\langle 2^\Sigma, \subseteq \rangle$, i.e. the powerset of Σ. This set of *interpretations* of Σ is denoted by $\mathcal{I}_\Sigma$. Following the framework of approximation theory, we will, however, interpret programs in the bilattice $B_\Sigma = \mathcal{I}_\Sigma^2$. In keeping with the intuitions presented in section 2.2, for such a pair (X, Y), the interpretation X can be seen as representing an *under*estimate of the set of true atoms, while Y represents an *over*estimate. Or, to put it another way, X contains all atoms which are *certainly* true, while Y contains atoms which are *possibly* true. These intuitions lead naturally to the following definition of the truth value of a

propositional formula.

Definition 4.1. Let ϕ, ψ be propositional formula in an alphabet Σ, a an atom of Σ and let $(X, Y) \in B_\Sigma$. We define

—$H_{(X,Y)}(a) = \mathbf{t}$ if $a \in X$; otherwise,$H_{(X,Y)}(a) = \mathbf{f}$;

—$H_{(X,Y)}(\phi \wedge \psi) = \mathbf{t}$ if $H_{(X,Y)}(\phi) = \mathbf{t}$ and $H_{(X,Y)}(\psi) = \mathbf{t}$; otherwise, $H_{(X,Y)}(\phi \wedge \psi) = \mathbf{f}$;

—$H_{(X,Y)}(\phi \vee \psi) = \mathbf{t}$ if $H_{(X,Y)}(\phi) = \mathbf{t}$ or $H_{(X,Y)}(\psi) = \mathbf{t}$; otherwise, $H_{(X,Y)}(\phi \vee \psi) = \mathbf{f}$;

—$H_{(X,Y)}(\neg\phi) = \mathbf{t}$ if $H_{(Y,X)}(\phi) = \mathbf{f}$; otherwise, $H_{(X,Y)}(\neg\phi) = \mathbf{f}$.

Note that to evaluate the negation of a formula $\neg\phi$ in a pair (X, Y), we actually evaluate ϕ in (Y, X). Indeed, the negation of a formula will be certain if the formula itself is not possible and vice versa. Using this definition, we can now define the following operator on B_Σ.

Definition 4.2. Let P be a logic program with an alphabet Σ. The operator $\mathcal{T}_P$ on B_Σ is defined as:

$$\mathcal{T}_P(X, Y) = (U_P(X, Y), U_P(Y, X)),$$

with $U_P(X, Y) = \{p \in \Sigma \mid \exists r \in P : head(r) = p, H_{(X,Y)}(body(r)) = \mathbf{t}\}$.

When restricted to consistent pairs of interpretation, this operator $\mathcal{T}_P$ is the well known 3-valued Fitting operator [Fitting 1985]. In Denecker et al. [2000], $\mathcal{T}_P$ is shown to be a symmetric approximation. Furthermore, it can be used to define most of the "popular" semantics for logic programs: the operator which maps an interpretation X to $U_P(X, X)$ is the well known (two-valued) T_P-operator [Lloyd 1987]; the partial stable operator of $\mathcal{T}_P$ is the Gelfond-Lifschitz operator $\mathcal{GL}$ [Van Gelder et al. 1991]. Fixpoints of T_P are *supported models* of P, the least fixpoint of $\mathcal{T}_P$ is the Kripke-Kleene model of P, fixpoints of $\mathcal{GL}$ are (four-valued) stable models of P and its least fixpoint is the well-founded model of P.

4.1.2 *Stratification.* Our discussion of the stratification of logic programs will be based on the dependecies between atoms, which are expressed by a logic program. These induce the following partial order on the alphabet of the program.

Definition 4.3. Let P be a logic program with alphabet Σ. The *dependency order* $\leq_{dep}$ on Σ is defined as: for all $p, q \in \Sigma$:

$$p \leq_{dep} q \qquad iff \qquad \exists r \in P : q = head(r), p \in body(r).$$

To illustrate this definition, consider the following small program:

$$E = \left\{ \begin{array}{l} p \leftarrow \neg q, \neg r. \\ q \leftarrow \neg p, \neg r. \\ s \leftarrow p, q. \end{array} \right\}.$$

The dependency order of this program can be graphically represented by the following picture:
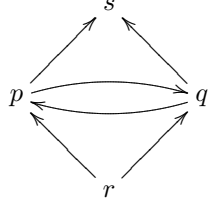
In other words, $r \leq_{dep} p, r \leq_{dep} q, p \leq_{dep} q, q \leq_{dep} p, s \leq_{dep} p$ and $s \leq_{dep} q$.

Based on this dependency order, the concept of a *splitting* of the alphabet of a logic program can be defined.

Definition 4.4. Let P be a logic program with alphabet Σ. A *splitting* of P is a partition $(\Sigma_i)_{i \in I}$ of Σ, such that the well-founded order $\preceq$ on I agrees with the dependency order $\leq_{dep}$ of P, i.e. if $p \leq_{dep} q$, $p \in \Sigma_i$ and $q \in \Sigma_j$, then $i \preceq j$.

For instance, the following partition is a splitting of the program E: $\Sigma_0 = \{r\}$, $\Sigma_1 = \{p, q\}$ and $\Sigma_2 = \{s\}$ (with the index set I being the totally ordered set $\{0, 1, 2\}$).

If $(\Sigma_i)_{i \in I}$ is a partition of a logic program P with alphabet Σ, the product lattice $\otimes_{i \in I} 2^{\Sigma_i}$ is clearly isomorphic to the powerset 2^{Σ}. We can therefore view the operator $\mathcal{T}_P$ of such a program as being an operator on the bilattice of this product lattice, instead of on the original lattice B_{Σ}. Moreover, if such a partition is a splitting of a logic program P, the $\mathcal{T}_P$-operator on this product lattice is stratifiable.

THEOREM 4.5. *Let P be a logic program and let $(\Sigma_i)_{i \in I}$ be a splitting of this program. Then the operator $\mathcal{T}_P$ on the bilattice of the product lattice $\bigotimes_{i \in I} 2^{\Sigma_i}$ is stratifiable.*

PROOF. Let $\Sigma_j \in S$ and $(X, Y), (X', Y') \in B_{\Sigma}$, such that $X|_{\preceq i} = X'|_{\preceq i}$ and $Y|_{\preceq i} = Y'|_{\preceq i}$. It suffices to show that for each clause with an atom from Σ_j in its head, $H_{(X,Y)}(body(c)) = H_{(X',Y')}(body(c))$. By definition 4.4, this is trivially so. □

By theorem 3.11, this theorem implies that, for a stratifiable program P, it is possible to stratify the operators $T_P, \mathcal{T}_P$ and $\mathcal{GL}$. In other words, it is possible to split logic programs w.r.t. the supported model, Kripke-Kleene, stable model and well-founded semantics. Moreover, the supported, Kripke-Kleene, stable and well-founded models of P can be computed from, respectively, the supported, Kripke-Kleene, stable and well-founded models of the components of the operator $\mathcal{T}_P$.

In order to be able to perform this construction in practice, however, we also need a more precise characterization of these components. We will now show how to construct new logic programs from the original program, such that these components correspond to an operator associated with these new programs. First, we will define the restriction of a program to a subset of its alphabet.

Definition 4.6. Let P be a logic program with a splitting $(\Sigma_i)_{i \in I}$. For each $i \in I$, the program P_i consists of all clauses which have an atom from Σ_i in their head.

In the case of our example, the program E is partitioned in $\{E_0, E_1, E_2\}$ with $E_0 = \{\}$, $E_1 = \{p \leftarrow \neg q, \neg r. \; q \leftarrow \neg p, \neg r.\}$ and $E_2 = \{s \leftarrow p, q.\}$.

If P has a splitting $(\Sigma_i)_{i \in I}$, then clearly such a program P_i contains, by definition, only atoms from $\bigcup_{j \preceq i} \Sigma_j$. When given a pair (U, V) of interpretations of $\bigcup_{j \prec i} \Sigma_j$, we can therefore construct a program containing only atoms from Σ_i by replacing each other atom by its truth-value according to (U, V).

Definition 4.7. Let P be a logic program with a splitting $(\Sigma_i)_{i \in I}$. For each $i \in I$ and $(U, V) \in B_\Sigma|_{\prec i}$, we define $P_i \langle (U, V) \rangle$ as the new logic program P', which results from replacing each literal l whose atom is in $\bigcup_{j \prec i} \Sigma_j$ by $H_{(U,V)}(\{l\})$.

Of course, one can further simplify such a program by removing all clauses containing $\mathbf{f}$ and just omitting all atoms $\mathbf{t}$. Programs constructed in this way are now precisely those which characterize the components of the operator $\mathcal{T}_P$.

THEOREM 4.8. *Let P be a logic program with a splitting $(\Sigma_i)_{i \in I}$. For each $i \in I$, $(U, V) \in B_\Sigma|_{\prec i}$ and $(A, B) \in B_{\Sigma_i}$:*

$$(\mathcal{T}_P)_i^{(U,V)}(A, B) = (U_{P_i \langle (U,V) \rangle}(A, B), U_{P_i \langle (V,U) \rangle}(B, A)).$$

PROOF. Let i, U, V, A and B be as above. Then because the order $\preceq$ on I agrees with the dependency order of P, $(\mathcal{T}_P)_i^{(U,V)}(A, B) = (A', B')$, with

$$A' = \{p \in \Sigma_i \mid \exists r \in P : head(r) = p, H_{(U \sqcup A, V \sqcup B)}(body(r)) = \mathbf{t}\};$$
$$B' = \{p \in \Sigma_i \mid \exists r \in P : head(r) = p, H_{(V \sqcup B, U \sqcup A)}(body(r)) = \mathbf{t}\}.$$

We will show that $A' = \mathcal{T}_{P_i \langle (U,V) \rangle}(A, B)$; the proof that $B' = \mathcal{T}_{P_i \langle (V,U) \rangle}(B, A)$ is similar. Let r be a clause of P, such that $head(r) \in \Sigma_i$. Then $H_{(U \sqcup A, V \sqcup B)}(body(r)) = \mathbf{t}$ iff $H_{(U,V)}(l) = \mathbf{t}$ for each literal l with an atom from $\bigcup_{j \prec i} \Sigma_j)$ and $H_{(A,B)}(l') = \mathbf{t}$ for each literal $'l$ with an atom from Σ_i. Because for each literal l with an atom from $\bigcup_{j \prec i} \Sigma_j)$, $H_{(U,V)}(l) = \mathbf{t}$ precisely iff l was replaced by $\mathbf{t}$ in $P_i \langle (U, V) \rangle$, this is in turn equivalent to $H_{(A,B)}(r \langle (U, V) \rangle) = \mathbf{t}$ (by $r \langle (U, V) \rangle$ we denote the clause which replaces r in $P_i \langle (U, V) \rangle$). $\square$

It is worth noting that this theorem implies that a component $(\mathcal{T}_P)_i^{(U,V)}$ is, in contrast to the operator $\mathcal{T}_P$ itself, not necessarily exact.

With this final theorem, we have all which is needed to incrementally compute the various fixpoints of the operator $\mathcal{T}_P$. We will illustrate this process by computing the well-founded model of our example program E. Recall that this program is partitioned into the programs $E_0 = \{\}$, $E_1 = \{p \leftarrow \neg q, \neg r. \; q \leftarrow \neg p, \neg r.\}$ and $E_2 = \{s \leftarrow p, q.\}$. The well-founded model of E_0 is $(\{\}, \{\})$. Replacing the atom r in E_1 by its truth-value according to this interpretation, yields the new program $E_1 \langle (\{\}, \{\}) \rangle = \{p \leftarrow \neg q. \; q \leftarrow \neg p.\}$. The well-founded model of this program is $(\{\}, \{p, q\})$. Replacing the atoms p and q in E_2 by their truth-value according to the pair of interpretations $(\{\}, \{p, q\})$, gives the new program $E_2' = E_2 \langle (\{\}, \{p, q\}) \rangle = \{\}$. Replacing these by their truth-value according to the pair of interpretations $(\{p, q\}, \{\})$, gives the new program $E_2'' = E_2 \langle (\{p, q\}, \{\}) \rangle = \{s\}$. The well-founded fixpoint of $(U_{E_2'}, U_{E_2''})$ is $(\{\}, \{s\})$. Therefore, the well-founded model of the entire program E is $(\{\} \cup \{\} \cup \{\}, \{\} \cup \{p, q\} \cup \{s\}) = (\{\}, \{p, q, s\})$.

Of course, it also possible to apply these results to more complicated programs. Consider for instance the following program in the natural numbers:

$$Even = \left\{ \begin{array}{c} even(0). \\ odd(X+1) \leftarrow even(X). \\ even(X+1) \leftarrow odd(X). \end{array} \right\}$$

which can be seen as an abbreviation of the infitine propositional logic program:

$$
\begin{aligned}
even(0). \\
odd(1) &\leftarrow even(0). \\
even(1) &\leftarrow odd(0). \\
odd(2) &\leftarrow even(1). \\
even(2) &\leftarrow odd(1). \\
&\vdots
\end{aligned}
$$

Clearly, the operator $\mathcal{T}_{Even}$ is stratifiable w.r.t. to the partition

$$(\{even(n), odd(n)\})_{n \in \mathbb{N}}$$

(using the standard order on the natural numbers) of the alphabet $\{even(n) \mid n \in \mathbb{N}\} \cup \{odd(n) \mid n \in \mathbb{N}\}$. The component $(\mathcal{T}_{Even})_0$ of this operator corresponds to the program $Even_0 = \{even(0).\}$, which has $\{even(0)\}$ as its only fixpoint. Let $n \in \mathbb{N}$ and $U_n = \{even(i) \mid i < n, i \text{ is even}\} \cup \{odd(i) \mid i < n, i \text{ is odd}\}$. Clearly, if n is even, the component $(T_{Even})_n^{(U_n,U_n)}$ corresponds to the program $\{even(n).\}$, while if n is odd $(T_{Even})_n^{(U_n,U_n)}$ corresponds to the program $\{odd(n).\}$. This proves that the supported, Kripke-Kleene, stable and well-founded models of the program $Even$ all contain precisely those atoms $even(n)$ for which n is an even natural number and those atoms $odd(n)$ for which n is an odd natural number.

4.1.3 *Related work.* Lifschitz and Turner [1994] proved a splitting theorem for logic programs under the stable model semantics; similar results were indepently obtained by Eiter et al. [1997]. In one respect, these results extend ours, in the sense that they apply to logic programs with an extended syntax, also called *answer set programs*, in which disjunction in the head of clauses is allowed and in which two kinds of negation (negation-as-failure and classical negation) can be used. While our results could easily be extended to incorporate the two negations, the extension to disjunction in the head is less straightforward. However, the fact that the stable model semantics for disjunctive logic programs can also be characterized as a fixpoint semantics [Leone et al. 1995], seems to suggest that our approach could be used to obtain similar results for this extended syntax as well. In future work, we plan to investigate this further. Current work into extending approximation theory to also capture the semantics of this kind of programs [Pelov 2004], makes this possibility seem even more interesting.

In another respect, our results are more general than those of Lifschitz and Turner. When considering only programs in the syntax described here, our results generalize their results to include the supported model, Kripke-Kleene and

well-founded semantics as well. This makes our results also applicable to extensions of logic programming which, unlike answer set programming, are not based on the stable model semantics. One example in the context of deductive databases is datalog which is based on well-founded semantics. Another example is the formalism of ID-logic [Denecker 2000; Denecker and Ternovska 2004]. This is an extension of classical logic with non-monotonic inductive definitions, the semantics of which is given by the well-founded semantics. Our stratifiability results give insight in modularity and compositionality aspects in both logics.

While the results discussed above are, as far as we know, the ones most similar to ours, there are a number of other works which should be also be mentioned. Extending work from Verbaeten et al. [2000], Denecker and Ternovska [2004] recently discussed a number of modularity results for the aforementioned ID-logic. The main difference with our work, is that these results are aimed at supporting syntactical transformations on a *predicate* level, whereas we have considered propositional progams.

In order to further motivate and explain the well-founded model semantics, Przymusinski [1998] defined the *dynamic stratification* of a program. The level of an atom in this stratificiation is based on the number of iterations it takes the Gelfond-Lifschitz operator to determine the truth-value of this atom.[1] As such, this stratification precisely mimics the computation of the well-founded model and is, therefore, the tightest possible stratification of a program under the well-founded semantics. However, as there exist no syntactic criteria which can be used to determine whether a certain stratification is the dynamic stratification of a program – in fact, the only way of deciding this is by actually constructing the well-founded model of the program – this concept cannot be used to perform the kind of static, upfront splitting which is our goal.

4.2 Auto-epistemic logic

In this section, we will first describe the syntax of auto-epistemic logic and give a brief overview, based on Denecker et al. [2003], of how a number of different semantics for this logic can be defined using concepts from approximation theory. Then, we will follow the previously outlined methodology to prove concrete splitting results for this logic.

4.2.1 *Syntax and Semantics.* Let $\mathcal{L}$ be the language of propositional logic based on a set of atoms Σ. Extending this language with a modal operator K, gives a language $\mathcal{L}_K$ of modal propositional logic. An auto-epistemic theory is a set of fomulas in this language $\mathcal{L}_K$. For such a formula ϕ, the subset of Σ containing all atoms which appear in ϕ, is denoted by $At(\phi)$; atoms which appear in ϕ at least once outside the scope of the model operator K are called *objective* atoms of ϕ and the set of all objective atoms of ϕ is denoted by $At_O(\phi)$. A *modal subformula* is a formula of the form $K(\psi)$, with ψ a formula.

To illustrate, consider the following example:

$$F = \{\phi_1 = p \vee \neg Kp \ ; \ \ \phi_2 = K(p \vee q) \vee q\}$$

---

[1]To be a bit more precise, p belongs to level Σ_i iff i is the minimal j for which $\mathcal{GL} \uparrow j = (I, J)$ and either $p \in I$ or $p \notin J$.

. The objective atoms $At_O(\phi_2)$ of ϕ_2 are $\{q\}$, while the atoms $At(\phi_2)$ are $\{p,q\}$. The formula $K(p \lor q)$ is a modal subformula of ϕ_2.

An *interpretation* is a subset of the alphabet Σ. The set of all interpretations of Σ is denoted by $\mathcal{I}_\Sigma$, i.e. $\mathcal{I}_\Sigma = 2^\Sigma$. A *possible world structure* is a set of interpretations, i.e. the set of all possible world structures $\mathcal{W}_\Sigma$ is defined as $2^{\mathcal{I}_\Sigma}$. Intuitively, a possible world structure sums up all "situations" which are possible. It therefore makes sense to order these according to inverse set inclusion to get a *knowledge order* $\leq_k$, i.e. for two possible world stuctures Q, Q', $Q \leq_k Q'$ iff $Q \supseteq Q'$. Indeed, if a possible world structure contains *more* possibilities, it actually contains *less* knowledge.

In case of the example, the following picture shows a part of the lattice $\mathcal{W}_{At(F)}$:
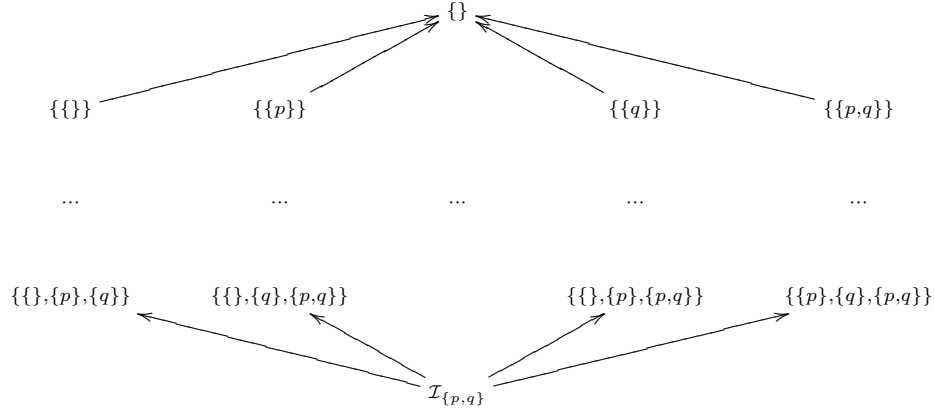
Following Denecker et al. [2003], we will define the semantics of an auto-epistemic theory by an operator on the bilattice $\mathcal{B}_\Sigma = \mathcal{W}_\Sigma^2$. An element (P, S) of $\mathcal{B}_\Sigma$ is known as a *belief pair*. In a belief pair (P, S), with $P \leq_k S$, P can be viewed as describing what must *certainly* be known, i.e. as giving an *under*estimate of what is known, while S can be viewed as denoting what might *possibly* be known, i.e. as giving an *over*estimate. Based on this intuition, there are two ways of estimating the truth of modal formulas according to (P, S): we can either be *conservative*, i.e. assume a formula is false unless we are sure it must be true, or we can be *liberal*, i.e. assume it is true unless we are sure it must be false. To conservatively estimate the truth of a formula $K\phi$ according to (P, S), we simply have to check whether ϕ is surely known, i.e. whether ϕ is known in the underestimate P. To conservatively estimate the truth of a formula $\neg K\phi$, on the other hand, we need to determine whether ϕ is definitely unknown; this will be the case if ϕ cannot possibly be known, i.e. if ϕ is not known in the overestimate S. The following definition extends these intuitions to reach a conservative estimate of the truth of arbitrary formulas. Note that the objective atoms of such a formula are simply interpreted by an interpretation $X \in \mathcal{I}_\Sigma$.

Definition 4.9. For each $(P, S) \in \mathcal{B}_\Sigma, X \in \mathcal{I}_\Sigma, a \in \Sigma$ and formulas ϕ, ϕ_1 and ϕ_2, we inductively define $\mathcal{H}_{(P,S),X}$ as:

—$\mathcal{H}_{(P,S),X}(a) = \mathbf{t}$ iff $a \in X$ for each atom a;
—$\mathcal{H}_{(P,S),X}(\varphi_1 \land \varphi_2) = \mathbf{t}$, iff $\mathcal{H}_{(P,S),X}(\varphi_1) = \mathbf{t}$ and $\mathcal{H}_{(P,S),X}(\varphi_2) = \mathbf{t}$;

—$\mathcal{H}_{(P,S),X}(\varphi_1 \vee \varphi_2) = \mathbf{t}$, iff $\mathcal{H}_{(P,S),X}(\varphi_1) = \mathbf{t}$ or $\mathcal{H}_{(P,S),X}(\varphi_2) = \mathbf{t}$;
—$\mathcal{H}_{(P,S),X}(\neg\varphi) = \neg\mathcal{H}_{(S,P),X}(\varphi)$;
—$\mathcal{H}_{(P,S),X}(K\varphi) = \mathbf{t}$ iff $\mathcal{H}_{(P,S),Y}(\varphi) = \mathbf{t}$ for all $Y \in P$.

It is worth noting that an evaluation $\mathcal{H}_{(Q,Q),X}(\phi)$, i.e. one in which all that might possibly be known is also surely known, corresponds to the standard S_5 evaluation [Meyer and van der Hoek 1995]. Note also that the evaluation $\mathcal{H}_{(P,S),X}(K\phi)$ of a modal subformula $K\phi$ depends only on (P,S) and not on X. We sometimes emphasize this by writing $\mathcal{H}_{(P,S),\cdot}(K\phi)$. Similarly, $\mathcal{H}_{(P,S),X}(\phi)$ of an objective formula ϕ depends only on X and we sometimes write $\mathcal{H}_{(\cdot,\cdot),X}(\phi)$.

As mentioned above, it is also possible to liberally estimate the truth of modal subformulas. Intuitively, we can do this by assuming that everything which might be known, *is* in fact known and that everything which might be unknown, i.e. which is not surely known, is in fact unknown. As such, to liberally estimate the truth of a formula according to a pair (P,S), it suffices to treat S as though it were describing what we surely know and P as though it were describing what we might know. In others words, it suffices to simply switch the roles of P and S, i.e. $\mathcal{H}_{(S,P),\cdot}$ provides a liberal estimate of the truth of modal formulas.

These two ways of evaluating formulas can be used to derive a new, more precise belief pair (P',S') from an original pair (P,S). First, we will focus on constructing the new overestimate S'. As S' needs to overestimate knowledge, it needs to contain as few interpretations as possible. This means that S' should consist of only those interpretations, which manage to satisfy the theory even if the truth of its modal subformulas is underestimated. So, $S' = \{X \in \mathcal{I}_\Sigma \mid \forall \phi \in T : \mathcal{H}_{(P,S),X}(\phi) = \mathbf{t}\}$. Conversely, to construct the new underestimate P', we need as many interpretations as possible. This means that P' should contain all interpretations which satisfy the theory, when liberally evaluating its modal subformulas. So, $P' = \{X \in \mathcal{I}_\Sigma \mid \forall \phi \in T : \mathcal{H}_{(S,P),X}(\phi) = \mathbf{t}\}$. These intuitions motivate the following definition of the operator $\mathcal{D}_T$ on $\mathcal{B}_\Sigma$:

$$\mathcal{D}_T(P,S) = (\mathcal{D}_T^u(S,P), \mathcal{D}_T^u(P,S)),$$

with $\mathcal{D}_T^u(P,S) = \{X \in \mathcal{I}_\Sigma \mid \forall \phi \in T : \mathcal{H}_{(P,S),X}(\phi) = \mathbf{t}\}$.

This operator is an approximation [Denecker et al. 2003]. Moreover, since

$$\mathcal{D}_T^1(P,S) = \mathcal{D}_T^u(S,P) = \mathcal{D}_T^2(S,P),$$

and

$$\mathcal{D}_T^2(P,S) = \mathcal{D}_T^u(P,S) = \mathcal{D}_T^1(S,P),$$

it is by definition symmetric and therefore approximates a unique operator on $\mathcal{W}_\Sigma$, namely the operator D_T [Moore 1984], which maps each Q to $\mathcal{D}_T^u(Q,Q)$. As shown in Denecker et al. [2003], these operators define a family of semantics for a theory T:

—fixpoints of D_T are *expansions* of T [Moore 1984],
—fixpoints of $\mathcal{D}_T$ are *partial expansions* of T [Denecker et al. 1998],
—the least fixpoint of $\mathcal{D}_T$ is the *Kripke-Kleene fixpoint* of T [Denecker et al. 1998],
—fixpoints of $C_{\mathcal{D}_T}^\downarrow$ are *extensions* [Denecker et al. 2003] of T,

—fixpoints of $\mathcal{C}_{\mathcal{D}_T}$ are *partial extensions* [Denecker et al. 2003] of T, and

—the least fixpoint of $\mathcal{C}_{\mathcal{D}_T}$ is the *well-founded model* of T [Denecker et al. 2003].

These various dialects of auto-epistemic logic differ in their treatment of "ungrounded" expansions [Konolige 1987], i.e. expansions which arize from cyclicities such as $Kp \to p$.

When calculating the models of a theory, it is often useful to split the calculation of $\mathcal{D}_T^u(P, S)$ into two separate steps: In a first step, we evaluate each modal subformula of T according to (P, S) and in a second step we then compute all models of the resulting propositional theory. To formalize this, we introduce the following notation: for each formula ϕ and $(P, S) \in \mathcal{B}_\Sigma$, the formula $\phi\langle P, S \rangle$ is inductively defined as:

—$a\langle P, S \rangle = a$ for each atom a;

—$(\varphi_1 \wedge \varphi_2)\langle P, S \rangle = \varphi_1\langle P, S \rangle \wedge \varphi_2\langle P, S \rangle$;

—$(\varphi_1 \vee \varphi_2)\langle P, S \rangle = \varphi_1\langle P, S \rangle \vee \varphi_2\langle P, S \rangle$;

—$(\neg\varphi)\langle P, S \rangle = \neg(\varphi\langle S, P \rangle)$;

—$(K\varphi)\langle P, S \rangle = \mathcal{H}_{(P,S),\cdot}(K\varphi)$.

For a theory T, we denote $\{\phi\langle P, S \rangle \mid \phi \in T\}$ by $T\langle P, S \rangle$. Because clearly

$$\mathcal{H}_{(\cdot,\cdot),X}(T\langle P, S \rangle) = \mathbf{t} \text{ iff } \mathcal{H}_{(P,S),X}(T) = \mathbf{t},$$

it is the case that, for each $(P, S) \in \mathcal{B}_\Sigma$, $\mathcal{D}_T^u(P, S)$ is the set $Mod(T\langle P, S \rangle)$ of all classical models of the propositional theory $T\langle P, S \rangle$.

To illustrate, we will construct the Kripke-Kleene model of our example theory $F = \{p \vee \neg Kp; \ K(p \vee q) \vee q\}$. This computation starts at the least precise element $(\mathcal{I}_{\{p,q\}}, \{\})$ of $\mathcal{B}_{\{p,q\}}$. We first construct the new underestimate $\mathcal{D}_F^u(\{\}, \mathcal{I}_{\{p,q\}})$. It is easy to see that

$$\mathcal{H}_{(\{\},\mathcal{I}_{\{p,q\}}),\cdot}(\neg Kp) = \neg\mathcal{H}_{(\mathcal{I}_{\{p,q\}},\{\}),\cdot}(Kp) = \neg\mathbf{f} = \mathbf{t},$$

and

$$\mathcal{H}_{(\{\},\mathcal{I}_{\{p,q\}}),\cdot}(K(p \vee q)) = \mathbf{t}.$$

Therefore, $F\langle\{\}, \mathcal{I}_{\{p,q\}}\rangle = \{p \vee \mathbf{t}; \ q \vee \mathbf{t}\}$ and $\mathcal{D}_F^u(\{\}, \mathcal{I}_{\{p,q\}}) = \mathcal{I}_{\{p,q\}}$. Now, to compute the new overestimate $\mathcal{D}_F^u(\mathcal{I}_{\{p,q\}}, \{\})$, we note that

$$\mathcal{H}_{(\mathcal{I}_{\{p,q\}},\{\}),\cdot}(\neg Kp) = \neg\mathcal{H}_{(\{\},\mathcal{I}_{\{p,q\}}),\cdot}(Kp) = \neg\mathbf{t} = \mathbf{f},$$

and

$$\mathcal{H}_{(\mathcal{I}_{\{p,q\}},\cdot),\cdot}(K(p \vee q)) = \mathbf{f}.$$

Therefore, $F\langle(\mathcal{I}_{\{p,q\}}, \{\})\rangle = \{p \vee \mathbf{f}; \ q \vee \mathbf{f}\}$ and $\mathcal{D}_F^u(\mathcal{I}_{\{p,q\}}, \{\}) = \{\{p, q\}\}$. So, $\mathcal{D}_T(\mathcal{I}_{\{p,q\}}, \{\}) = (\mathcal{I}_{\{p,q\}}, \{\{p, q\}\})$.

To compute $\mathcal{D}_F^u(\{p, q\}, \mathcal{I}_{\{p,q\}})$, we note that it is still the case that

$$\mathcal{H}_{(\cdot,\mathcal{I}_{\{p,q\}}),\cdot}(\neg Kp) = \mathcal{H}_{(\{\{p,q\}\},\cdot),\cdot}(K(p \vee q) = \mathbf{t}.$$

So, $\mathcal{D}_F^u(\{\{p, q\}\}, \mathcal{I}_{\{p,q\}}) = \mathcal{I}_{\{p,q\}}$. Similary,

$$\mathcal{H}_{(\cdot,\{\{p,q\}\}),\cdot}(\neg Kp) = \mathcal{H}_{(\mathcal{I}_{\{p,q\}},\cdot),\cdot}(\neg Kp) = \mathbf{f}.$$

So, $\mathcal{D}_F^u(\mathcal{I}_{\{p,q\}}, \{\{p, q\}\}) = \{\{p, q\}\}$. Therefore, $(\mathcal{I}_{\{p,q\}}, \{\{p, q\}\})$ is the least fixpoint of $\mathcal{D}_F$, i.e. the Kripke-Kleene model of F.

4.2.2 *An intermediate operator.* Let $(\Sigma_i)_{i \in I}$ be a partition of the alphabet Σ, with $\langle I, \preceq \rangle$ a well-founded index set. For an interpretation $X \in \mathcal{I}_\Sigma$, we denote the intersection $X \cap \Sigma_i$ by $X|_{\Sigma_i}$. For a possible world structure Q, $\{X|_{\Sigma_i} \mid X \in Q\}$ is denoted by $Q|_{\Sigma_i}$.

In the previous section, we defined the semantics of auto-epistemic logic in terms of an operator on the bilattice $\mathcal{B}_\Sigma = \mathcal{W}_\Sigma^2$. However, for our purpose of stratifying auto-epistemic theories, we are interested in the bilattice $\tilde{\mathcal{B}}_\Sigma$ of the product lattice $\tilde{\mathcal{W}}_\Sigma = \bigotimes_{i \in I} \mathcal{W}_{\Sigma_i}$. An element of this product lattice consists of a number of possible interpretations for each level Σ_i. As such, if we choose for each Σ_i one of its interpretations, the union of these "chosen" interpretations interprets the entire alphabet Σ. Therefore, the set of all possible ways of choosing one interpretation for each Σ_i, determines a set of possible interpretations for Σ, i.e. an element of $\mathcal{W}_\Sigma$. More formally, we define:

$$\kappa : \tilde{\mathcal{W}}_\Sigma \to \mathcal{W}_\Sigma : \tilde{Q} \mapsto \{\bigcup_{i \in I} S(i) \mid S \in \otimes_{i \in I} \tilde{Q}(i)\}.$$

Similarly, $\tilde{\mathcal{B}}_\Sigma$ can be mapped to $\mathcal{B}_\Sigma$ by the function $\overline{\kappa}$, which maps each $(\tilde{P}, \tilde{S}) \in \tilde{\mathcal{B}}_\Sigma$ to $(\kappa(\tilde{P}), \kappa(\tilde{S}))$.

This function κ is a homomorphism, preserving the knowledge order $\leq_k$. However, it is not an isomorphism. Let us first demonstrate that it is not surjective. Unlike $\mathcal{W}_\Sigma$, elements of $\tilde{\mathcal{W}}_\Sigma$ cannot express that an interpretation for a level Σ_i is possible in combination with a *certain* interpretation for another level Σ_j, but *not* with a different interpretation for Σ_j. For instance, if we split the alphabet $\{p, q\}$ of our example F into $\Sigma_0 = \{p\}$ and $\Sigma_1 = \{q\}$, the element $\{\{p, q\}, \{\}\}$ of $\mathcal{W}_\Sigma$ is not in $\kappa(\tilde{\mathcal{W}}_\Sigma)$, because it expresses that $\{p\}$ is only a possible interpretation for Σ_0 when Σ_1 is interpreted by $\{q\}$ and not when Σ_1 is interpreted by $\{\}$. To make this more precise, we introduce the following concept of a possible world structure Q being *independent* w.r.t. a certain partition $(\Sigma_i)_{i \in I}$ of the alphabet. Intuitively, this is the case if, whenever an interpretation X_i for Σ_i is possible in combination with some interpretation Y_j for Σ_j, then X_i is also possible in combination with every other interpretation Y_j' for Σ_j appearing in Q.

Definition 4.10. A possible world structure $Q \in \mathcal{W}_\Sigma$ is *independent* w.r.t. a partition $(\Sigma_i)_{i \in I}$ of its alphabet iff for all possible worlds $X, Y \in Q$ and for each $i \in I$, $(X|_{\Sigma_i} \cup \bigcup_{j \neq i} Y|_{\Sigma_j}) \in Q$.

Clearly, we can now characterise the image of κ as follows.

PROPOSITION 4.11.

$$\kappa(\tilde{\mathcal{W}}_\Sigma) = \{Q \in \mathcal{W}_\Sigma \mid Q \text{ is independent}\}.$$

We now restrict our attention to a class of theories whose models are all independent possible world structures.

Definition 4.12. An auto-epistemic theory T is *stratifiable* w.r.t. a partition $(\Sigma_i)_{i \in I}$ of its alphabet and a well-founded order $\preceq$ on I, if there exists a partition $\{T_i\}_{i \in I}$ of T such that for each $i \in I$ and $\phi \in T_i$: $At_O(\phi) \subseteq \Sigma_i$ and $At(\phi) \subseteq \bigcup_{j \preceq i} \Sigma_j$.

To illustrate, our example theory F is stratifiable w.r.t. the partition $\Sigma_0 = \{p\}$, $\Sigma_1 = \{q\}$ of its alphabet $\{p, q\}$ and the corresponding partition of F is $F_0 = \{p \vee \neg Kp\}$, $F_1 = \{K(p \vee q) \vee q\}$.

Clearly, for a stratifiable theory, the evaluation $\mathcal{H}_{(P,S),X}(\phi)$ of a formula $\phi \in T_i$ only depends on the value of (P, S) in strata $j \preceq i$ and that of X in stratum i.

PROPOSITION 4.13. *Let T be a stratifiable auto-epistemic theory. Let $i \in I$ and $\phi \in T_i$. Then for each $(P, S), (P', S') \in \mathcal{B}_\Sigma$ and $X, X' \in \mathcal{I}_\Sigma$, such that $X|_{\Sigma_i} = X|_{\Sigma_i}$ and $\forall j \preceq i, P|_{\Sigma_j} = P'|_{\Sigma_j}$ and $S|_{\Sigma_j} = S'|_{\Sigma_j}$, $\mathcal{H}_{(P,S),X}(\phi) = \mathcal{H}_{(P',S'),X'}(\phi)$.*

This proposition can now be used to show that only independent belief pairs are relevant for the operator $\mathcal{D}_T$.

PROPOSITION 4.14. *Let T be a stratifiable auto-epistemic theory. Then each $\mathcal{D}_T(P, S)$ is independent.*

PROOF. Let $(P, S) \in \mathcal{B}_\Sigma$ and $X, Y \in \mathcal{D}_T^u(P, S)$. By proposition 4.13, for each $Z \in \mathcal{I}_\Sigma$, such that, for some $i \in I$, $Z|_{\Sigma_i} = X|_{\Sigma_i}$ and, $\forall j \neq i$, $Z|_{\Sigma_j} = Y|_{\Sigma_j}$, $Z \in \mathcal{D}_T^u(P, S)$. Therefore $\mathcal{D}_T^u(P, S)$ and $\mathcal{D}_T(P, S)$ are both independent. □

This result suggests that the fact that κ is not surjective should not pose any problems, because we can simply forget about possible world structures of $\mathcal{W}_\Sigma$ that do not correspond to elements of our product lattice $\tilde{\mathcal{W}}_\Sigma$. However, there is another difference between the lattices $\tilde{\mathcal{W}}_\Sigma$ and $\mathcal{W}_\Sigma$, that needs to be taken into account. Indeed, besides not being surjective, κ is also not an embedding of $\tilde{\mathcal{W}}_\Sigma$ into $\mathcal{W}_\Sigma$. Concretely, $\tilde{\mathcal{W}}_\Sigma$ contains multiple "copies" of the empty set, that is, for any $\tilde{Q} \in \tilde{\mathcal{W}}_\Sigma$, as soon as for some $i \in I$, $\tilde{Q}(i) = \{\}$, it is the case that $\kappa(\tilde{Q}) = \{\}$.

Let us introduce some notation and terminology. We call an possible world structure $Q \in \mathcal{W}_\Sigma$ *consistent* if $Q \neq \{\}$; the set of all consistent Q is denoted by $\mathcal{W}_\Sigma^c$. A belief pair (P, S) is called consistent if both P and S are consistent; the set of all consistent belief pairs is denoted $\mathcal{B}_\Sigma^c$. Similarly, $\tilde{Q} \in \tilde{\mathcal{B}}_\Sigma$ is called consistent if $\kappa(\tilde{Q}) \neq \{\}$ and the set of all consistent $\tilde{Q}$ is denoted as $\tilde{\mathcal{W}}_\Sigma^c$. Finally, a belief pair $(\tilde{P}, \tilde{S})$ is called consistent if both $\kappa(\tilde{P}) \neq \{\}$ and $\kappa(\tilde{S}) \neq \{\}$ and we denote the set of all consistent belief pairs as $\tilde{\mathcal{B}}_\Sigma^c$.

We will often need to eliminate inconsistent possible world structures from our considerations. Intuitively, the reason for this is that, when constructing a stratification, we need every stratum i to be completely independent of all strata j for which $j \succ i$. However, if an inconsistency occurs at level j, then this could affect the way in which a lower level i is interpreted, because it will eliminate *all* possible worlds. Mathematically, this problem manifest itself by the fact that the equality $\kappa(\tilde{Q})|_{\preceq i} = \kappa(\tilde{Q}|_{\preceq i})$ only holds for consistent possible world structures $\tilde{Q}$.

We now summarise the properties of κ discussed so far.

PROPOSITION 4.15. *The function κ has the following properties.*

(1) *κ is order preserving;*

(2) *κ is an embedding of $\tilde{\mathcal{W}}_\Sigma^c$ into $\mathcal{W}_\Sigma^c$ and an isomorphism between $\tilde{\mathcal{W}}_\Sigma^c$ and the set of all independent possible world structrures in $\mathcal{W}_\Sigma^c$;*

(3) *For all consistent $\tilde{Q}$, $\kappa(\tilde{Q}|_{\preceq i}) = \kappa(\tilde{Q})|_{\preceq i}$.*

Because of the differences between the lattices $\tilde{\mathcal{B}}_\Sigma$ and $\mathcal{B}_\Sigma$ outlined above, we cannot directly stratify the operator $\mathcal{D}_T$. Instead, we will define an intermediate operator $\tilde{\mathcal{D}}_T$ on $\tilde{\mathcal{B}}_\Sigma$, which is stratifiable by construction and whose fixpoints are related to the fixpoints of the $\mathcal{D}_T$. We define this operator $\tilde{\mathcal{D}}_T$ in such a way that, for any belief pair $(\tilde{P}, \tilde{S}) \in \tilde{\mathcal{B}}_\Sigma$, the ith level of $\tilde{\mathcal{D}}_T(\tilde{P}, \tilde{S})$ will be constructed using only the theory T_i and the restriction $(\tilde{P}, \tilde{S})|_{\preceq i}$ of $(\tilde{P}, \tilde{S})$.

Definition 4.16. Let T be a stratifiable auto-epistemic theory. Let $(\tilde{P}, \tilde{S})$ be in $\tilde{\mathcal{B}}_\Sigma$. We define $\tilde{\mathcal{D}}_T^u(\tilde{P}, \tilde{S}) = \tilde{Q}$, with for each $i \in I$:

$$\tilde{Q}(i) = \{X \in \mathcal{I}_{\Sigma_i} \mid \forall \phi \in T_i : \mathcal{H}_{\overline{\kappa}(\tilde{P}|_{\preceq i}, \tilde{S}|_{\preceq i}), X}(\phi) = \mathbf{t}\}.$$

Furthermore, $\tilde{\mathcal{D}}_T(\tilde{P}, \tilde{S}) = (\tilde{\mathcal{D}}_T^u(\tilde{S}, \tilde{P}), \tilde{\mathcal{D}}_T^u(\tilde{P}, \tilde{S}))$ and $\tilde{\mathcal{D}}_T(\tilde{Q}) = \tilde{\mathcal{D}}_T^u(\tilde{Q}, \tilde{Q})$.

Observe that we could equivalently define the ith level of $\tilde{\mathcal{D}}_T^u(\tilde{P}, \tilde{S})$ as the set $Mod(T_i\langle \overline{\kappa}(\tilde{P}|_{\preceq i}, \tilde{S}|_{\preceq i})\rangle)$ of all classical models of the propositional theory $T_i\langle \overline{\kappa}(\tilde{P}|_{\preceq i}, \tilde{S}|_{\preceq i})\rangle$.

Let us now first show that, like its counterpart $\mathcal{D}_T$, this operator is also an approximation.

PROPOSITION 4.17. *Let T be a stratifiable auto-epistemic theory. Then $\tilde{\mathcal{D}}_T$ is an approximation.*

PROOF. Let $(\tilde{P}, \tilde{S}), (\tilde{P}', \tilde{S}') \in \tilde{\mathcal{B}}_\Sigma$, such that $(\tilde{P}, \tilde{S}) \leq_p (\tilde{P}', \tilde{S}')$. Because then $(\tilde{S}, \tilde{P}) \geq (\tilde{S}', \tilde{P}')$, we only need to show that $\tilde{\mathcal{D}}_T^u(\tilde{P}, \tilde{S}) \geq_\otimes \tilde{\mathcal{D}}_T^u(\tilde{P}', \tilde{S}')$. As κ is clearly order-preserving, $\overline{\kappa}(\tilde{P}, \tilde{S}) \leq_p \overline{\kappa}(\tilde{P}', \tilde{S}')$. From Denecker et al. [2003], we know this implies that for each ϕ of T and X in $\mathcal{I}_\Sigma$, if $\mathcal{H}_{\overline{\kappa}(\tilde{P}, \tilde{S}), X} = \mathbf{t}$ then $\mathcal{H}_{\overline{\kappa}(\tilde{P}', \tilde{S}'), X} = \mathbf{t}$. Hence, $\tilde{\mathcal{D}}_T^u(\tilde{P}, \tilde{S})(i) \subseteq \tilde{\mathcal{D}}_T^u(\tilde{P}', \tilde{S}')(i)$. $\square$

Now, we will relate the consistent fixpoints of $\tilde{\mathcal{D}}_T$ to those of $\mathcal{D}_T$.

PROPOSITION 4.18. *For all consistent $(\tilde{P}, \tilde{S}) \in \tilde{\mathcal{B}}_\Sigma^c$, $\overline{\kappa}(\tilde{\mathcal{D}}_T(\tilde{P}, \tilde{S})) = \mathcal{D}_T(\overline{\kappa}(\tilde{P}, \tilde{S}))$.*

PROOF. Let $(\tilde{P}, \tilde{S}) \in \tilde{\mathcal{B}}_\Sigma^c$. By symmetry of the operators $\tilde{\mathcal{D}}_T$ and $\mathcal{D}_T$, it suffices to show that $\kappa(\tilde{\mathcal{D}}_T^u(\tilde{P}, \tilde{S})) = \mathcal{D}_T^u(\kappa(\tilde{P}), \kappa(\tilde{S}))$. Because for $i \neq j$, the objective atoms of T_i and T_j are disjoint, it is a trivial property of propositional logic that $Mod(T\langle \overline{\kappa}(P, S)\rangle)$ consists precisely of all worlds of the form $\cup_i X_i$ for which $X_i \in Mod(T_i\langle \kappa(\tilde{P}), \kappa(\tilde{S})\rangle)$. As such, if we let $\tilde{Q}$ be the element of $\tilde{\mathcal{B}}_\Sigma$ that maps every $i \in I$ to $Mod(T_i\langle \kappa(\tilde{P}), \kappa(\tilde{S})\rangle)$, we have that $\kappa(\tilde{Q}) = \mathcal{D}_T^u(\kappa(\tilde{P}), \kappa(\tilde{S}))$. It therefore suffices to show that $\tilde{Q} = \tilde{\mathcal{D}}_T^u(\tilde{P}, \tilde{S})$, that is, that for all $i \in I$, $T_i\langle \kappa(\tilde{P}), \kappa(\tilde{S})\rangle = T_i\langle \kappa(\tilde{P}|_{\preceq i}), \kappa(\tilde{S}|_{\preceq i})\rangle$. Because T_i contains only modal literals in the alphabet $\cup_{j \preceq i} \Sigma_j$, we already have that $T_i\langle \kappa(\tilde{P}), \kappa(\tilde{S})\rangle = T_i\langle \kappa(\tilde{P})|_{\preceq i}, \kappa(\tilde{S})|_{\preceq i}\rangle$. Because $(\tilde{P}, \tilde{S})$ is consistent, we also have that $(\kappa(\tilde{P})|_{\preceq i}, \kappa(\tilde{S})|_{\preceq i}) = (\kappa(\tilde{P}|_{\preceq i}), \kappa(\tilde{S}|_{\preceq i}))$, which proves the result. $\square$

We already know that κ is an isomorphism between $\tilde{\mathcal{B}}_\Sigma^c$ and the set of all consistent, independent belief pairs in $\mathcal{B}_\Sigma$, and that all consistent fixpoints of $\mathcal{D}_T$ belong are also independent. Therefore, the above Proposition 4.18 now directly implies the following result.

PROPOSITION 4.19. *The set of all $\kappa(\tilde{P}, \tilde{S})$ for which $(\tilde{P}, \tilde{S})$ is a consistent fixpoint of $\tilde{\mathcal{D}}_T$ is equal to the set of all consistent fixpoints of $\mathcal{D}_T$.*

A similar correspondence also holds for the consistent stable fixpoints of these two operators. Our proof of this depends on the following result.

PROPOSITION 4.20. *For all consistent $\tilde{S} \in \tilde{\mathcal{B}}_\Sigma^c$, $\kappa(C_{\tilde{\mathcal{D}}_T}^\downarrow(\tilde{S})) = C_{\mathcal{D}_T}^\downarrow(\kappa(\tilde{S}))$.*

PROOF. Recall that the operator $C_{\mathcal{D}_T}^\downarrow$ is defined as mapping each S to $lfp(\mathcal{D}_T^u(\cdot, S))$ and, similarly, $C_{\tilde{\mathcal{D}}_T}^\downarrow$ maps each $\tilde{S}$ to $lfp(\tilde{\mathcal{D}}_T^u(\cdot, \tilde{S}))$. Let $\tilde{S} \in \tilde{\mathcal{W}}_\Sigma^c$ and $S = \kappa(\tilde{S})$. The values $C_{\mathcal{D}_T}^\downarrow(S)$ and $C_{\tilde{\mathcal{D}}_T}^\downarrow(\tilde{S})$ can be constructed as the limit of, respectively, the ascending sequences $(Q_i)_{0 \le i}$ and $(\tilde{Q}_i)_{0 \le i}$, defined as follows: Q_0 is the bottom element $\mathcal{I}_\Sigma$ of $\mathcal{W}_\Sigma$ and for every $i > 0$, $Q_i = \mathcal{D}_T^u(Q_{i-1}, S)$; similarly, $\tilde{Q}_0$ is the bottom element of $\tilde{\mathcal{W}}_\Sigma$, that is, for all $i \in I$, $\tilde{Q}_0(i) = \mathcal{I}_{\Sigma_i}$, and for all $i > 0$, $\tilde{Q}_i = \tilde{\mathcal{D}}_T^u(\tilde{Q}_{i-1}, \tilde{S})$. It is well-known that there must now exist $m, \tilde{m} \in \mathbb{N}$, for which $Q_{m+1} = Q_m$ and $\tilde{Q}_{\tilde{m}+1} = \tilde{Q}_{\tilde{m}}$, and that $Q_m = lfp(\mathcal{D}_T^u(\cdot, S))$ and $\tilde{Q}_{\tilde{m}} = lfp(\tilde{\mathcal{D}}_T^u(\cdot, \tilde{S}))$. It therefore suffices to show that, for all $n \in \mathbb{N}$, $Q_n = \kappa(\tilde{Q}_n)$.

We prove this by induction over n. For the base case, it is clear that $\kappa(\tilde{Q}_0) = Q_0$. Now, suppose that the equality holds for n. Then $\tilde{Q}_{n+1} = \tilde{\mathcal{D}}_T^u(\tilde{Q}_n, \tilde{S})$, which we must prove equal to $Q_{n+1} = \mathcal{D}_T^u(Q_n, S)$. By the induction hypothesis, this last expression is equal to $\mathcal{D}_T^u(\kappa(\tilde{Q}_n), \kappa(\tilde{S}))$. We distinguish two cases. First, let us assume that $\tilde{Q}_n$ is consistent. By Proposition 4.18, it is then the case that $\kappa(\tilde{\mathcal{D}}_T^u(\tilde{Q}_n, \tilde{S})) = \mathcal{D}_T^u(\kappa(\tilde{Q}_n, \tilde{S}))$, which is precisely what needs to be proven. Second, assume that $\kappa(\tilde{Q}_n) = Q_n = \{\}$. Because $\tilde{Q}_{n+1} \ge_k \tilde{Q}_n$ and κ is order-preserving, $\kappa(\tilde{Q}_{n+1}) = \{\}$. Moreover, because also $Q_{n+1} \ge_k Q_n = \{\}$, we have that $Q_{n+1} = \{\}$, which means that, here too, we get the desired equality $\kappa(\tilde{Q}_{n+1}) = Q_{n+1}$. □

Together with the fact that κ is an isomorphism between $\tilde{\mathcal{W}}_\Sigma^c$ and the set of all consistent, independent belief pairs, this result now directly implies the following correspondence between consistent stable fixpoints.

PROPOSITION 4.21. *The set of all $\kappa(\tilde{P}, \tilde{S})$ for which $(\tilde{P}, \tilde{S})$ is a consistent stable fixpoint of $\tilde{\mathcal{D}}_T$ is equal to the set of all consistent stable fixpoints of $\tilde{\mathcal{D}}_T$.*

We can now summarise the content of Propositions 4.19 and 4.21 in the following theorem.

THEOREM 4.22. *Let T be a stratifiable theory. The set of all $\overline{\kappa}(\tilde{P}, \tilde{S})$ for which $(\tilde{P}, \tilde{S})$ is a consistent fixpoint or a consistent stable fixpoint of $\tilde{\mathcal{D}}_T$ is equal to the set of all (P, S), for which (P, S) is a consistent fixpoint or, respectively, a consistent stable fixpoint of $\mathcal{D}_T$.*

Let us now consider a special class of theories, for which it is clear that the inconsistent possible world set $\{\}$ can never play a relevant role.

Definition 4.23. A theory T is *permaconsistent* if every propositional theory T' that can be constructed from T by replacing all occurrences of modal literals by **t** or **f** is consistent.

Observe that, contrary to what the above definition might suggest, we do not actually need to check *every* assignment in order to determine whether a theory is permaconsistent. Indeed, it suffices to only consider the worst case assignment, in which every positive occurrence of a modal literal is replaced by **f** and every negative occurrence is replaced by **t**. Moreover, we can also check permaconsistency for every stratum separately, because, for a stratifiable theory T, T is permaconsistent iff for every $i \in I$, T_i is permaconsistent.

Clearly, for a permaconsistent theory T, all fixpoints or stable fixpoints of $\mathcal{D}_T$ and $\tilde{\mathcal{D}}_T$ must be consistent. As such, Theorem 4.22 implies that the fixpoints and stable fixpoints of $\mathcal{D}_T$ and $\tilde{\mathcal{D}}_T$ coincide, which of course implies that also the least fixpoint and well-founded fixpoint of these two operators coincide. In summary, we obtain the following result.

THEOREM 4.24. *Let T be a stratifiable and permaconsistent theory. The set of all $\overline{\kappa}(\tilde{P}, \tilde{S})$ for which $(\tilde{P}, \tilde{S})$ is a fixpoint, the Kripke-Kleene fixpoint, a stable fixpoint, or the well-founded fixpoint of $\tilde{\mathcal{D}}_T$ is equal to the set consisting of, respectively, all fixpoints, the Kripke-Kleene fixpoint, all stable fixpoints, or the well-founded fixpoint of $\mathcal{D}_T$.*

It is obvious that this property does not hold for theories that are not permaconsistent. We illustrate this by the following example. Let T be the theory $\{\neg p; q \wedge \neg Kp\}$. Clearly, T is stratifiable with respect to the partition $\Sigma_0 = \{p\}$, $\Sigma_1 = \{q\}$ of its alphabet, but T is not permaconsistent. The operator $\mathcal{D}_T$ now has as its Kripke-Kleene fixpoint the pair $(\{\{q\}\}, \{\})$, for which it is indeed the case that $Mod(T\langle \cdot, \{\{q\}\}\rangle) = Mod(\{\neg p; q \wedge \mathbf{t}\}) = \{\{q\}\}$ and $Mod(T\langle \cdot, \{\}\rangle) = Mod(\{\neg p; q \wedge \mathbf{f}\}) = \{\}$. The Kripke-Kleene fixpoint of $\tilde{\mathcal{D}}_T$, on the other hand, is the exact pair $(\tilde{Q}, \tilde{Q})$ for which $\tilde{Q}(0) = \{\{\}\}$ and $\tilde{Q}(1) = Mod(q \wedge \mathbf{t}) = \{\{q\}\}$. We have that $\overline{\kappa}(\tilde{Q}, \tilde{Q}) = (\{\{q\}\}, \{\{q\}\}) \neq (\{\{q\}\}, \{\})$.

4.2.3 *Stratification.* Using the correspondence between fixpoints of $\tilde{\mathcal{D}}_T$ and $\mathcal{D}_T$ from the previous section, we can now proceed to analyse $\mathcal{D}_T$ by stratifying $\tilde{\mathcal{D}}_T$. Clearly, $\tilde{\mathcal{D}}_T$ is by construction stratfiable. Therefore, we can incrementally construct the various models of a stratifiable theory from the components of $\tilde{\mathcal{D}}_T$. These components themselves can in turn be constructed by replacing certain parts of T by their truth value according to a partial pair of interpretations $(\tilde{U}, \tilde{V}) \in \tilde{\mathcal{B}}_\Sigma|_{\prec i}$. Before showing this for all stratifiable theories, we will first deal only with the following, more restricted class of theories.

Definition 4.25. A theory T is *modally separated* w.r.t. to a partition $(\Sigma_i)_{i \in I}$ of its alphabet iff there exists a corresponding partition $(T_i)_{i \in I}$ of T, such that for each $i \in I$ and $\phi \in T_i$

—$At_O(\phi) \subseteq \Sigma_i$,
—for each modal subformula $K\psi$ of ϕ, either $At(\psi) \subseteq \Sigma_i$ or $At(\psi) \subseteq \bigcup_{j \prec i} \Sigma_j$.

Clearly, modally separated theories are by definition stratifiable. The fact that each modal subformula of a level T_i of a modally separated theory T contains

either only atoms from Σ_i or only atoms from a strictly lower level, makes it easy to construct the components of its $\tilde{\mathcal{D}}_T$-operator. Replacing all modal subformulae of a level T_i which contain only atoms from a strictly lower level $j \prec i$, by their truth-value according to a partial belief pair $(\tilde{U}, \tilde{V}) \in \mathcal{B}_\Sigma|_{\prec i}$ results in a "conservative theory" T^c, while replacing these subformulae by their truth-value according to $(\tilde{V}, \tilde{U})$ yields a "liberal theory" T^l. The pair $(\mathcal{D}^u_{T^l}, \mathcal{D}^u_{T^c})$ is then precisely the component $(\tilde{\mathcal{D}}_T)^{(\tilde{U}, \tilde{V})}_i$ of $\tilde{\mathcal{D}}_T$.

To make this more precise, we inductively define the following transformation $\phi\langle U, V\rangle_i$ of a formula $\phi \in T_i$, given a partial belief pair $(\tilde{U}, \tilde{V}) \in \tilde{\mathcal{B}}_\Sigma|_{\prec i}$:

—$a\langle \tilde{U}, \tilde{V}\rangle_i = a$ for each atom a;

—$(\varphi_1 \wedge \varphi_2)\langle \tilde{U}, \tilde{V}\rangle_i = \varphi_1\langle \tilde{U}, \tilde{V}\rangle_i \wedge \varphi_2\langle \tilde{U}, \tilde{V}\rangle_i$;

—$(\varphi_1 \vee \varphi_2)\langle \tilde{U}, \tilde{V}\rangle_i = \varphi_1\langle \tilde{U}, \tilde{V}\rangle_i \vee \varphi_2\langle \tilde{U}, \tilde{V}\rangle_i$;

—$(\neg\varphi)\langle \tilde{U}, \tilde{V}\rangle_i = \neg(\varphi\langle \tilde{V}, \tilde{U}\rangle_i)$;

—$(K\varphi)\langle \tilde{U}, \tilde{V}\rangle_i = \begin{cases} \mathcal{H}_{(\tilde{U}, \tilde{V}),\cdot}(K\varphi) & \text{if } At(\varphi) \subseteq \bigcup_{j \prec i} \Sigma_j; \\ K(\varphi) & \text{if } At(\varphi) \subseteq \Sigma_i. \end{cases}$

Note that this transformation $\phi\langle \tilde{U}, \tilde{V}\rangle_i$ is identical to the transformation $\phi\langle P, S\rangle$ defined earlier, except for the fact that in this case, we only replace modal sub-formulae with atoms from $\bigcup_{j \prec i}$ and leave modal subformulae with atoms from Σ_i untouched.

From the various definitions, it is now clear that the components $(\tilde{\mathcal{D}}_T)^{(\tilde{U}, \tilde{V})}_i$ of the $\tilde{\mathcal{D}}_T$-operator of a modally separated theory T can be constructed as follows:

PROPOSITION 4.26. *Let T be a modally separated theory. Let $i \in I$, $(\tilde{U}, \tilde{V}) \in \tilde{\mathcal{B}}_\Sigma|_{\prec i}$ and $(\tilde{P}_i, \tilde{S}_i) \in \tilde{\mathcal{B}}_{\Sigma_i}$. Then:*

$$(\tilde{\mathcal{D}}_T)^{(\tilde{U}, \tilde{V})}_i(\tilde{P}_i, \tilde{S}_i) = (\mathcal{D}^u_{T_i\langle \tilde{V}, \tilde{U}\rangle_i}(\tilde{S}_i, \tilde{P}_i), \mathcal{D}^u_{T_i\langle \tilde{U}, \tilde{V}\rangle_i}(\tilde{P}_i, \tilde{S}_i)).$$

Now, all that remains is to characterize the components of stratifiable theories which are not modally separated. It turns out that for each stratifiable theory T, there exists a modally separated theory T', which is equivalent to ϕ w.r.t. evaluation in independent possible world structures. To simplify the proof of this statement, we recall that each formula ϕ can be written in an equivalent form ϕ' such that each modal subformula of ϕ' is of the form $K(a_1 \vee \cdots \vee a_m)$, with each a_i a literal. This result is well-known for S_5 semantics and can — using the same transformation — be shown to also hold for all semantics considered here[2].

PROPOSITION 4.27. *Let (P, S) be an independent element of $\mathcal{B}_\Sigma$. Let $i \in I$, $b_1, \ldots, b_n$ literals with atoms from Σ_i and $c_1, \ldots, c_m$ literals with atoms from $\bigcup_{j \prec i} \Sigma_j$.*

---

[2]To show this, it suffices to show that each step of this transformation preserves the value of the evalation $\mathcal{H}_{(P,S),X}(\phi)$. For all steps corresponding to properties of (three-valued) propositional logic, this is trivial. The step of transforming a formula $K(K(\phi))$ to $K(\phi)$ also trivially satisfies this requirement. All that remains to be shown, therefore, is that $\mathcal{H}_{(P,\cdot),\cdot}(K(\phi \wedge \psi)) = \mathcal{H}_{(P,\cdot),\cdot}(K(\phi) \wedge K(\psi))$. By definition, $\mathcal{H}_{(P,\cdot),\cdot}(K(\phi \wedge \psi)) = \mathbf{t}$ iff $\forall X \in P : \mathcal{H}_{(\cdot,\cdot),X}(\phi) = \mathbf{t}$ and $\mathcal{H}_{(\cdot,\cdot),X}(\phi) = \mathbf{t}$, which in turn is equivalent to $\forall X \in P : \mathcal{H}_{(\cdot,\cdot),X}(K(\phi)) = \mathbf{t}$ and $\forall X \in P : \mathcal{H}_{(\cdot,\cdot),X}(K(\psi)) = \mathbf{t}$.

Then

$$\mathcal{H}_{(P,S),\cdot}(K(\bigvee_{j=1..n} b_j \vee \bigvee_{j=1..m} c_j)) = \mathcal{H}_{(P,S),\cdot}(K(\bigvee_{j=1..n} b_j) \vee K(\bigvee_{j=1..m} c_j)).$$

PROOF. By definition,

$$\mathcal{H}_{(P,S),\cdot}(K(\bigvee_{j=1..n} b_j \vee \bigvee_{j=1..m} c_j)) = \mathbf{t}$$

iff

$$\forall X \in P : \mathcal{H}_{(\cdot,\cdot),X}(\bigvee_{j=1..n} b_j \vee \bigvee_{j=1..m} c_j) = \mathbf{t}.$$

This is equivalent to $\forall X \in P$, $\mathcal{H}_{(\cdot,\cdot),X}(\bigvee_{j=1..n} b_j) = \mathbf{t}$ or $\mathcal{H}_{(\cdot,\cdot),X}(\bigvee_{j=1..m} c_j) = \mathbf{t}$. Because P is independent, it contains all possible combinations $X|_{\cup_{j \prec i} \Sigma_j} \cup Y|_{\Sigma_i} \cup Z|_{\cup_{j \not\prec i} \Sigma_j}$, with $X, Y, Z \in P$. Therefore the previous statement is in turn equivalent to for each $X, Y \in P$, $\mathcal{H}_{(\cdot,\cdot),X|_{\Sigma_i}}(\bigvee_{j=1..n} b_j) = \mathbf{t}$ or $\mathcal{H}_{(\cdot,\cdot),Y|_{\cup_{j \prec i} \Sigma_j}}(\bigvee_{j=1..m} c_j)$, which proves the result. □

The modally separated formula corresponding to a formula ϕ will be denoted by $[\phi]$. In the case of our example $F = \{p \vee \neg Kp;\ K(p \vee q) \vee q\}$, the modally separated theory $[F] = \{p \vee \neg Kp;\ K(p) \vee K(q) \vee q\}$ is equivalent to F w.r.t. evaluation in $\overline{\kappa}(\tilde{\mathcal{W}}_{\{p,q\}})$. This results now allows us to characterize the components of all stratifiable theories.

THEOREM 4.28. *Let* $i \in I$, $(\tilde{U}, \tilde{V}) \in \tilde{\mathcal{B}}_\Sigma|_{\prec i}$ *and* $(\tilde{P}_i, \tilde{S}_i) \in \tilde{\mathcal{B}}_{\Sigma_i}$. *Then:*

$$(\tilde{\mathcal{D}}_T)_i^{(\tilde{U}, \tilde{V})}(\tilde{P}_i, \tilde{S}_i) = (\mathcal{D}_{[T_i]\langle(\tilde{V}, \tilde{U})\rangle}^u(\tilde{S}_i, \tilde{P}_i), \mathcal{D}_{[T_i]\langle(\tilde{U}, \tilde{V})\rangle}^u(\tilde{P}_i, \tilde{S}_i)).$$

Putting all of this together, we arrive at the following results. Let us first assume a permaconsistent theory. Combining Theorem 4.24 with the above Theorem 4.28, we get that an element (P, S) of $\mathcal{B}_\Sigma$ is a (partial) expansion, (partial) extension, Kripke-Kleene fixpoint or well-founded model of a stratifiable theory T iff there exists a $(\tilde{P}, \tilde{S}) \in \tilde{\mathcal{B}}_\Sigma$, such that $\overline{\kappa}(\tilde{P}, \tilde{S}) = (P, S)$ and for all $i \in I$, $(\tilde{P}, \tilde{S})(i)$ is, respectively, a (partial) expansion, (partial) extension, Kripke-Kleene fixpoint or well-founded model of $[T_i]\langle(\tilde{P}, \tilde{S})|_{\prec i}\rangle$. For theories which are not permaconsistent, we only get these correspondences for *consistent* belief pairs, as shown by Theorem 4.22.

Using these results, we can incrementally construct the models of a stratifiable permaconsistent theory under each of these semantics. To illustrate, we will use these results to incrementally compute the Kripke-Kleene model of our permaconsistent example F, which we previously partitioned into $F_0 = \{p \vee \neg Kp\}$ and $F_1 = \{K(p \vee q) \vee q\}$. The Kripke-Kleene model of F_0 is $(\{\{\}, \{p\}\}, \{\{p\}\})$. Let F_1' be $[F_1]\langle(\{\{p\}\}, \{\{\}, \{p\}\})\rangle = \{\mathbf{t} \vee K(q) \vee q\}$ and let F_1'' be $[F_1]\langle(\{\{\}, \{p\}\}, \{\{p\}\})\rangle = \{\mathbf{f} \vee K(q) \vee q\}$. The least fixpoint of $(\mathcal{D}_{F_1'}^u, \mathcal{D}_{F_1''}^u)$ is $(\{\{\}, \{q\}\}, \{\{q\}\})$. Therefore, the Kripke-Kleene fixpoint of F is $(\mathcal{I}_{\{p,q\}}, \{\{p, q\}\})$. Of course, (partial) expansions, (partial) extensions and the well-founded model of F can be computed in a similar manner.

4.2.4 *Related work.* In Gelfond and Przymusinska [1992] and Niemelä and Rintanen [1994], it was shown that certain permaconsistent and modally separated auto-epistemic theories can be split under the semantics of expansions. We have both extended these results to other semantics for this logic and to a larger class of theories.

To give some intuition about the kind of theories our result can deal with, but previous work cannot, we will consider the following example (from Etherington [1988]): Suppose we would like to express that we suspect a certain person of murder if we know he had a motive and if it is possible that this person is a suspect and that he is guilty. This naturally leads to following formula:

$$K\,motive \wedge \neg K(\neg suspect \vee \neg guilty) \rightarrow suspect.$$

This formula is not modally separated w.r.t. the partition

$$\Sigma_0 = \{guilty, motive\}, \Sigma_1 = \{suspect\}$$

and, therefore, falls outside the scope of Gelfond et al.'s theorem. Our result, however, does cover this example and allows it to be split w.r.t. this partition. As we will discuss in the next section on default logic, there exists an important class of default expressions, called *semi-normal defaults*, which typically give rize to such statements.

4.3 Default logic

4.3.1 *Syntax and semantics.* Let $\mathcal{L}$ be the language of propositional logic for an alphabet Σ. A *default* d is a formula

$$\frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma}$$

with $\alpha, \beta_1, \ldots, \beta_n, \gamma$ formulas of $\mathcal{L}$. The formula γ is called the consequence $cons(d)$ of d. A *default theory* is a pair $\langle D, W \rangle$, with D a set of defaults and W formulas of $\mathcal{L}$.

Konolige [1987] suggested a transformation m from default logic to auto-epistemic logic, which was shown by Denecker et al. [2003] to capture the semantics of default logic. For simplicity, we will ignore the original formulation of the semantics of default logic and view this as being defined by the auto-epistemic theory $m(\langle D, W \rangle)$.

Definition 4.29. Let $\langle D, W \rangle$ be a default theory and let $d = \frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma}$ be a default in D. Then

$$m(d) = (K\alpha \wedge \neg K\neg\beta_1 \wedge \cdots \wedge \neg K\neg\beta_n \Rightarrow \gamma)$$

and

$$m(\langle D, W \rangle) = \{m(d) \mid d \in D\} \cup W.$$

A pair (P, S) of possible world structures is a (partial) expansion [Denecker et al. 2003], (partial) extension [Reiter 1980], Kripke-Kleene [Denecker et al. 2003] or well-founded model [Denecker et al. 2003] of a default theory $\langle D, W \rangle$ if it is, respectively, a (partial) expansion, (partial) extension, Kripke-Kleene or well-founded model of $m(\langle D, W \rangle)$. Of all these semantics, the semantics of extensions is the most common.

4.3.2   *Stratification.* We begin by defining the concept of a stratifiable default theory.

Definition 4.30. Let $\langle D, W \rangle$ be a default theory over an alphabet Σ. Let $(\Sigma_i)_{i \in I}$ be a partition of Σ. $\langle D, W \rangle$ is stratifiable over this partition, if there exists a partition $\langle D_i, W_i \rangle_{i \in I}$ of $\langle D, W \rangle$ such that:

—For each default d: if an atom of $cons(d)$ is in Σ_i, then $d \in D_i$,

—For each default d: all atoms of d are in $\bigcup_{j \preceq i} \Sigma_j$.

—For each $w \in W$, if w contains an atom $p \in \Sigma_i$, then $w \in W_i$.

The auto-epistemic theory corresponding to a stratifiable default theory is also stratifiable (according to definition 4.12).

THEOREM 4.31. *Let $\langle D, W \rangle$ be a stratifiable default theory. Then $m(\langle D, W \rangle)$ is a stratifiable auto-epistemic theory.*

PROOF. Let $\langle D, W \rangle$ be a default theory over an alphabet Σ and $(\Sigma_i)_{i \in I}$ a partition of Σ, such that $\langle D, W \rangle$ is stratifiable over this partition. Let $\langle D_i, W_i \rangle_{i \in I}$ be a partition of $\langle D, W \rangle$ satisfying the conditions of definition 4.30. We will show that the partition $(m(D_i) \cup W_i)_{i \in I}$ of $m(D, W)$ statisfies the conditions of definition 4.12.

Let $i \in I$. Each objective atoms in $m(D_i) \cup W_i$ is either a consequence of a default in D_i or appears in W_i. Hence, because of definition 4.30, each such atom is in Σ_i. All other atoms of $m(D_i) \cup W_i$ are, once again by definition, in $\bigcup_{j \preceq i} \Sigma_j$. □

By the results of the previous section, this proposition shows that we can also split default theories w.r.t. the semantics of constistent (partial) expansions and consistent (partial) extensions.

4.3.3   *Related work.* Turner [1996] proved splitting theorems for default logic under the semantics of consistent extensions. We extend these results to the semantics of consistent partial extensions and consistent (partial) expansions. Moreover, his results only apply to default theories $\langle D, W \rangle$ for which the auto-epistemic theory $m(\langle D, W \rangle)$ is modally separated. Our results therefore not only generalize previous results to other semantics, but also to a larger class of theories.

A typical example of a default which is not modally separated but which can be split using our results, is the example from Etherington [1988] concerning murder suspects. This can be formalized by the following default:

$$\frac{motive : suspect \land guilty}{suspect}.$$

In the previous section, we already presented the auto-epistemic formula resulting from applying the Konolige transformation to this default and showed that it was not modally stratified w.r.t. the partition

$$\Sigma_0 = \{guilty, motive\}, \Sigma_1 = \{suspect\}.$$

Therefore, Turner's theorem does not apply in this case, but our results do.

Defaults such as these are typical examples of so-called *semi-normal defaults*, i.e. defaults of the form

$$\frac{\alpha : \beta}{\gamma}$$

where β implies γ. This typically occurs because there is some formula δ, such that $\beta = \gamma \wedge \delta$. In such cases, the Konolige transformation will contain a formula $K(\neg\gamma \vee \neg\delta)$ and such defaults can therefore only be modally separated w.r.t. stratifications in which all atoms from both γ and δ belong to the same stratum. Our results, however, also allows stratifications in which (all or some) atoms from δ belong to a strictly lower stratum than the atoms from γ.

5. CONCLUSION

Stratification is, both theoretically and practically, an important concept in knowledge representation. We have studied this issue at a general, algebraic level by investigating stratification of *operators* and *approximations* (Section 3). This gave us a small but very useful set of theorems, which enabled us to easily and uniformly prove splitting results for all fixpoint semantics of logic programs, auto-epistemic logic and default logic (Section 4), thus generalizing existing results.

As such, the importance of the work presented here is threefold. Firstly, there are the concrete, applied results of Section 4 themselves. Secondly, there is the general, algebraic framework for the study of stratification, which can be applied to every formalism with a fixpoint semantics. Finally, on a more abstract level, our work also offers greater insight into the principles underlying various existing stratification results, as we are able to "look beyond" purely syntactical properties of a certain formalism.

REFERENCES

APT, K., BLAIR, H., AND WALKER, A. 1988. Towards a theory of Declarative Knowledge. In *Foundations of Deductive Databases and Logic Programming*, J. Minker, Ed. Morgan Kaufmann.

BARAL, C. AND SUBRAHMANIAN, V. 1991. Duality between alternative semantics of logic programs and nonmonotonic formalisms. In *International Workshop on Logic Programming and Nonmonotonic Reasoning*, A. Nerode, W. Marek, and V. Subrahmanian, Eds. MIT Press, Washington DC., 69–86.

DENECKER, M. 2000. Extending classical logic with inductive definitions. In *1st International Conference on Computational Logic (CL2000)*, J. Lloyd et al., Ed. Lecture Notes in Artificial Intelligence, vol. 1861. Springer, London, 703–717.

DENECKER, M., MAREK, V., AND TRUSZCZYNSKI, M. 1998. Fixpoint 3-valued semantics for autoepistemic logic. In *Proceedings of the 15th National Conference on Artificial Intelligence*. MIT Press / AAAI-Press, 840–845.

DENECKER, M., MAREK, V., AND TRUSZCZYNSKI, M. 2000. Approximating operators, stable operators, well-founded fixpoints and applications in non-monotonic reasoning. In *Logic-based Artificial Intelligence*. The Kluwer International Series in Engineering and Computer Science. Kluwer Academic Publishers, Boston, 127–144.

DENECKER, M., MAREK, V., AND TRUSZCZYNSKI, M. 2003. Uniform semantic treatment of default and autoepistemic logics. *Artificial Intelligence 143,* 1 (Jan.), 79–122.

DENECKER, M. AND TERNOVSKA, E. 2004. A logic of non-monotone inductive definitions and its modularity properties. In *LPNMR*. 47–60.

DIX, J. 1995. A classification theory of semantics normal logic programs: II. weak properties. *Fundamenta Informaticae XXII*, 257–288.

EITER, T., GOTTLOB, G., AND MANNILA, H. 1997. Disjunctive datalog. *ACM Transactions on Database Systems (TODS) 22*, 364–418.

ERDOĞAN, S. AND LIFSCHITZ, V. 2004. Definitions in Answer Set Programming. In *Proc. Logic Programming and Non Monotonic Reasoning, LPNMR'04*. LNAI, vol. 2923. Springer-Verlag, 185–197.

ETHERINGTON, D. 1988. *Reasoning with incomplete information*. Research notes in Artificial Intelligence. Morgan Kaufmann.

FITTING, M. 1985. A Kripke-Kleene Semantics for Logic Programs. *Journal of Logic Programming 2,* 4, 295–312.

FITTING, M. 1991. Bilattices and the semantics of logic programming. *Journal of Logic Programming 11*, 91–116.

GELFOND, M. 1987. On Stratified Autoepistemic Theories. In *Proc. of AAAI87*. Morgan Kaufman, 207–211.

GELFOND, M. AND PRZYMUSINSKA, H. 1992. On consistency and completeness of autoepistemic theories. *Fundamenta Informaticae 16,* 1, 59–92.

GINSBERG, M. 1988. Multivalued logics: a uniform approach to reasoning in artificial intelligence. *Computational Intelligence 4*, 265–316.

KONOLIGE, K. 1987. On the relation between default and autoepistemic logic. In *Readings in Nonmonotonic Reasoning*, M. L. Ginsberg, Ed. Kaufmann, Los Altos, CA, 195–226.

LEONE, N., RULLO, P., AND SCARCELLO, F. 1995. Declarative and fixpoints characterizations of disjunctive stable models. In *Proc. of International Logic Programming Ssymposium-ILPS'95*. MIT Press, 399–413.

LIFSCHITZ, V. AND TURNER, H. 1994. Splitting a logic program. In *Proceedings of the 11th International Conference on Logic Programming*. MIT Press, Cambridge, MA, USA, 23–37.

LLOYD, J. 1987. *Foundations of Logic Programming*. Springer-Verlag.

MEYER, J.-J.CH. AND VAN DER HOEK, W. 1995. *Epistemic Logic for Computer Science and Artificial Intelligence*. Cambridge University Press.

MOORE, R. C. 1984. Possible-world semantics for autoepistemic logic. In *Proceedings of the Non-Monotonic Reasoning Workshop*. AAAI, Mohonk, N.Y, 344–354.

PELOV, N. AND TRUSZCZYNSKI, M. 2004. Semantics of disjunctive programs with monotone aggregates - an operator-based approach. In *Proceedings of the 10th International Workshop on Non-Monotonic Reasoning (NMR 2004), Whistler, Canada, June 6-8, 2004, Proceedings*, J. P. Delgrande and T. Schaub, Eds. 327–334.

NIEMELÄ, I. AND RINTANEN, J. 1994. On the impact of stratification on the complexity of non-monotonic reasoning. *Journal of Applied Non-Classical Logics 4,* 2.

PRZYMUSINSKI, T. 1998. Every logic program has a natural stratification and an iterated least fixed point model. In *Proceedings of the 8th Symposium on Principles of Database Systems (PODS)*. 11–21.

REITER, R. 1980. A logic for default reasoning. *Artificial Intelligence 13,* 1–2, 81–132.

TARSKI, A. 1955. Lattice-theoretic fixpoint theorem and its applications. *Pacific Journal of Mathematics 5*, 285–309.

TURNER, H. 1996. Splitting a default theory. In *Proceedings of the 13th National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*. AAAI Press, 645–651.

VAN GELDER, A., ROSS, K., AND SCHLIPF, J. 1991. The Well-Founded Semantics for General Logic Programs. *Journal of the ACM 38,* 3, 620–650.

VENNEKENS, J., GILIS, D., AND DENECKER, M. 2004a. Splitting an operator: An algebraic modularity result and its application to auto-epistemic logic. In *Proceedings of International Workshop on Non-Monotonic Reasoning, Whistler, British Columbia, Canada*.

VENNEKENS, J., GILIS, D., AND DENECKER, M. 2004b. Splitting an operator: An algebraic modularity result and its applications to logic programming. In *Logic Programming, 20th International Conference, ICLP 2004, Proceedings*. Lecture Notes in Computer Science, vol. 3132. Springer, 195–209.

Verbaeten, S., Denecker, M., and Schreye, D. D. 2000. Compositionality of normal open logic programs. *Journal of Logic Programming 41*, 151–183.