decomposition of $m!$ may be found by the formula

$$t = \left[\frac{m}{p}\right]+\left[\frac{m}{p^2}\right]+\ldots+\left[\frac{m}{p^v}\right]+\ldots,$$

where $[a]$ is the integer part of a. Taking it into account, one sees that for the numerator this number is $(p^s-1)/(p-1)$ and for the denominator it is $(p^{s-1}-1)/(p-1)+(p^{s-1}-1)(p-1)/(p-1)$. The difference of these two numbers being equal to 1 proves that $\binom{d}{r} = \binom{p^s}{p^{s-1}}$ is not divisible by p^2. Now (9) shows that for some r also $p^2 \nmid n_r$. Therefore, in case (i), G_{c-1} satisfies the identity not holding in G_c if and only if R contains some element of the additive exponent p.

(ii) Secondly, assume that $d = c/2$ is divisible by two different primes p and q. In this case the greatest common divisor of numbers $n_2, \ldots, n_d$ is trivial. Indeed, $n_2 = d$ is divisible by both p and q, and $\binom{d}{r}$ for suitable choices of r is not divisible by either (this can be proved exactly as in (i)). Hence to obtain the conclusion on the greatest common divisor of $n_2, \ldots, n_d$ it remains to make use of (9). Applying formula (8), one sees that $\varrho u' \neq 0$ for any $\varrho \neq 0$ and therefore $\varrho u \equiv 0$ is not a law in G_{c-1} for any $0 \neq \varrho \in R$.

This completes the proof of the main theorem.

References

[1] K. K. Andreev, *Nilpotent groups and Lie algebras* (Russian), Algebra i Logika 7 (1969), 4–14.

[2] V. A. Artamonov, *Chain varieties of linear algebras* (Russian), Trudy Moscov. Mat. Obšč. 29 (1973), 51–77.

[3] Ju. A. Bahturin and A. Ju. Ol'sanskiĭ, *Soluble just-non-cross varieties of Lie algebras* (Russian), Mat. Sb. 100 (1976), 384–399.

[4] M. Hall, jr., *A basis for free Lie rings and higher commutators in free groups*, Proc. Amer. Math. Soc. 1 (1950), 575–581.

[5] L. G. Kovács, M. F. Newman, and P. F. Pentony, *Generating groups of nilpotent varieties*, Bull. Amer. Math. Soc. 74 (1968), 968–971.

[6] F. Levin, *On torsion free nilpotent varieties*, Comm. Pure Appl. Math. 26 (1973), 757–765.

[7] A. I. Malcev, *Algebraic systems*, Nauka, Moscow 1976.

[8] Hanna Neumann, *Varieties of groups*, Springer-Verlag, Berlin 1967.

[9] A. I. Širšov, *On free Lie rings* (Russian), Mat. Sb. 45 (1958), 113–122.

[10] M. R. Vaughan-Lee, *Generating groups of nilpotent varieties*, Bull. Austral. Math. Soc. 3 (1970), 145–154.

Presented to the Semester
Universal Algebra and Applications
(February 15 – June 9, 1978)

UNIVERSAL ALGEBRA AND APPLICATIONS
BANACH CENTER PUBLICATIONS, VOLUME 9
PWN–POLISH SCIENTIFIC PUBLISHERS
WARSAW 1982

A COMPLETENESS THEOREM IN THE MODAL LOGIC OF PROGRAMS*

KRISTER SEGERBERG

Åbo Academy, Åbo, Finland

As its name would seem to indicate, the modal logic of programs is, or can be viewed as, a generalization of classical modal logic. In spite of this fact there has been little interaction so far between the two fields. One wonders whether this is accidental, or whether there is a deeper explanation. For it may be that modal logicians and computer scientists are interested in rather different questions, or that already from the outset the modal logic of programs is headed for goals that lie beyond the limited territories of classical modal logic — the increased complexity of the former allows, even invites, such development, and application will probably demand it.

However this may be, it seems to the author that, at least in its present formative state, the modal logic of programs is truly a generalization of classical modal logic, and that the methods of the old discipline can be brought to bear on at least some of the basic problems in the emerging one. To give some substance to this claim we shall prove in this paper a completeness theorem of the kind of which there have been so many in modal logic. The theorem is interesting in its own right, but the main point is perhaps that the proof is achieved by a method that has been standard in modal logic for many years — the canonical models/filtrations technique, due originally to Dana Scott and others.

* The contents of this paper were presented in three lectures that the author delivered at the Stefan Banach International Mathematical Center in Warsaw on March 7 and 10, 1978. The author wishes to thank Professor H. Rasiowa, Dr T. Traczyk, and his other Polish hosts for a both pleasant and profitable visit to the Center, and Professor S. K. Thomason who was also there and discussed the lectures with him. He also wishes to record his thanks to Professors Rohit Parikh and, especially, Vaughan Pratt for valuable discussions and for the generous way in which they have shared with him their ideas on the modal logic of programs.

Part of the early work on the paper was supported by a grant (*stipendium för längre hunna vetenskapsidkare*) from the Academy of Finland.

In view of how diverse the audience of this paper is likely to be, an effort has been made to make the paper self-contained. For readers who would nevertheless like more background, the following comments are in order. The best reference for classical modal logic is presumably Lemmon [4]. Other references are Gabbay [1], Hughes & Cresswell [3] and Segerberg [11]. The originator of the modal logic of programs was evidently Vaughan Pratt (who nowadays prefers the more tractable term "dynamic logic"). His paper [6] contains some historical remarks and further bibliographical references. An idea of how dynamic logic might develop is given by his more recent paper [8]. The "extended abstract" Fischer & Ladner [2] is also a useful reference and was a source of inspiration for the present paper. Some further remarks on the writing of this paper will be found in Section 8 at the end of the paper.

1. Program modal languages

A (propositional) *program modal language* is determined by four sets which are supposed to be pairwise disjoint, viz.,

(i) a set Φ_0 of *propositional letters*,

(ii) a set Π_0 of *program letters*,

(iii) a set of *propositional operators*,

(iv) a set of *program operators*.

A given program modal language determines a set of program expressions and a set of formulas as follows. The set Π of *program expressions* is the smallest set Σ that satisfies the following conditions:

(i) $\Pi_0 \subseteq \Sigma$,

(ii) if φ is an n-ary program operator and $\alpha_0, \ldots, \alpha_{n-1} \in \Sigma$ then $\varphi(\alpha_0, \ldots, \alpha_{n-1}) \in \Sigma$.

The set Σ of *formulas* is the smallest set Σ that satisfies the following conditions:

(i) $\Phi_0 \subseteq \Sigma$,

(ii) if o is any n-ary propositional operator and $\boldsymbol{A}_0, \ldots, \boldsymbol{A}_{n-1} \in \Sigma$ then $o(\boldsymbol{A}_0, \ldots, \boldsymbol{A}_{n-1}) \in \Sigma$,

(iii) if $\alpha \in \Pi$ and $\boldsymbol{A} \in \Sigma$, then $[\alpha]\boldsymbol{A} \in \Sigma$.

Thus, for every program expression α, there is a unary propositional operator $[\alpha]$.

Although we have not done so here, it might be reasonable to add various conditions to the definition of program modal language. Instead, for the sake of concreteness, we shall assume, throughout most of the paper, as given a fixed program modal language that satisfies the following conditions:

(i) there are infinitely many propositional letters,

(ii) there are program letters, finitely or infinitely many,

(iii) the only propositional operators are the unary $\neg$ and the binary $\rightarrow$,

(iv) the only program operators are the binary $+$ and $\cdot$ and the unary $*$.

We use boldface Roman upper case letters $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ for formulas, and Greek lower case letters α, β for program expressions. Yet another typographical convention is to use italic lower case letters i, j, k, m, n, p for natural numbers, which we take to inlcude 0. Furthermore, we employ without explanation various simplifying conventions that are common. Thus we often drop parentheses, and we write $\boldsymbol{A} \rightarrow \boldsymbol{B}$ where protocol would demand $\rightarrow(\boldsymbol{A}, \boldsymbol{B})$. Non-primitive notation, such as $\wedge, \vee, \leftrightarrow$, will sometimes be used and may be thought of as abbreviational devices. For $+(\alpha, \beta)$ we write $\alpha+\beta$, for $\cdot(\alpha, \beta)$ we write, simply, $\alpha\beta$, and for $*(\alpha)$ we write α^*. We define

$$[\alpha]^0\boldsymbol{A} = \boldsymbol{A},$$

$$[\alpha]^{n+1}\boldsymbol{A} = [\alpha]([\alpha]^n\boldsymbol{A}).$$

In other words, $[\alpha]^n$ in a formula denotes a string of n $[\alpha]$'s. We speak of a program expression α in a formula or in a set of formulas if α occurs in the formula or in a formula in the set, respectively.

2. Semantics

By a *model* (suitable for the given language) we understand a triple $\langle U, R, V\rangle$ such that the following conditions hold:

(i) U is a set,

(ii) $R = \{R(\alpha)\}_{\alpha\in\Pi}$ is a family of binary relations on U; that is, if α is any program expression, then $R(\alpha) \subseteq U \times U$,

(iii) V is a function from Φ_0 to U; that is, if $\boldsymbol{P}$ is any propositional letter, then $V(\boldsymbol{P}) \subseteq U$.

Here U is called the *domain* and V the *valuation* of the model, while, for any α, $R(\alpha)$ is called the *accessibility relation corresponding to* α. Notice that the concept of model explicitly depends on the language (viz., on what program expressions there are).

We say that a model $\mathfrak{M} = \langle U, R, V\rangle$ is a *program model* if, for all α and β, it satisfies the following conditions:

(i) $R(\alpha+\beta) = R(\alpha) \cup R(\beta)$,

(ii) $R(\alpha\beta) = R(\alpha)|R(\beta)$,

(iii) $R(\alpha^*) = (R(\alpha))^*$.

Here we use the symbol | for relative product; that is, if S and T are any binary relations, then

$$S|T = \{\langle x, y\rangle: \exists z\, \langle x, z\rangle \in S \;\&\; \langle z, y\rangle \in T\}.$$

In (iii) we indulge in what is actually abuse of notation: while the first asterisk belongs to the object language, the second is a metalinguistic symbol representing the ancestral operation; that is, if S is any binary relation, then

$$S^* = \{S^n: n < \omega\},$$

where of course

$$S^0 = \{\langle x, x\rangle: x \text{ is in the field of } S\},$$

$$S^{n+1} = S|S^n.$$

Also the typographical shape + does double duty, in the object language as a program operator, in the metalanguage as representing addition between natural numbers. These ambiguities should cause no confusion.

Let $\mathfrak{M} = \langle U, R, V\rangle$ be any given model. The important concept of *truth at a point in* $\mathfrak{M}$ is recursively defined as follows (for $\mathfrak{M}\models_u \boldsymbol{A}$, read "$\boldsymbol{A}$ is true at u in $\mathfrak{M}$" or "$\boldsymbol{A}$ holds at u in $\mathfrak{M}$"). Suppose $u \in U$. Then:

(i) $\mathfrak{M}\models_u \boldsymbol{P}$ iff $u \in V(\boldsymbol{P})$, if $\boldsymbol{P}$ is a propositional letter,

(ii) $\mathfrak{M}\models_u \neg \boldsymbol{A}$ iff not $\mathfrak{M}\models_u \boldsymbol{A}$,

(iii) $\mathfrak{M}\models_u \boldsymbol{A} \to \boldsymbol{B}$ iff $\mathfrak{M}\models_u \boldsymbol{A}$ only if $\mathfrak{M}\models_u \boldsymbol{B}$,

(iv) $\mathfrak{M}\models_u [\alpha]\boldsymbol{A}$ iff, for all v, if $uR(\alpha)v$ then $\mathfrak{M}\models_v \boldsymbol{A}$.

We say that $\boldsymbol{A}$ is *true in* $\mathfrak{M}$ if, for all $u \in U$, $\mathfrak{M}\models_u \boldsymbol{A}$. If $\boldsymbol{A}$ is true in all models or in all program models, we say that $\boldsymbol{A}$ is *logically true*, respectively, *program logically true*.

Our analysis has led up to two obvious characterization problems: how to characterize the set of logically true formulas (not very interesting) and the set of program logically true formulas (interesting). Is the latter set axiomatizable? If so, is there a simple way of axiomatizing it? It is known from Fischer & Ladner [2] that the answer to the first question is affirmative. We shall show that also the (vaguely formulated) second question has an affirmative answer.

An aside: Students of modal logic will notice that the usual notions of frame and validity in a frame or in a class of frames would be as readily generalized as those of model and truth. It is not clear to the author how interesting such generalizations would be in the present context. It may be noticed, though, that the semantics of operators representing test programs (see, for example, Fischer & Ladner [2] or Pratt [7]) cannot be rendered in frame semantics: $\mathfrak{M}\models_u [\boldsymbol{A}?]\boldsymbol{B}$ holds if and only if either not $\mathfrak{M}\models_u \boldsymbol{A}$ or else $\mathfrak{M}\models_u \boldsymbol{B}$. This "difficulty" can of course be circumvented by introducing $[\boldsymbol{A}?]\boldsymbol{B}$ as an abbreviation of the formula $\boldsymbol{A} \to \boldsymbol{B}$. But the theoretical observation remains.

3. Syntax

By a *logic* (over the given language) we understand any set L of formulas that satisfies the following conditions:

(i) L contains every tautology (in the sense of ordinary two-valued propositional calculus),

(ii) L is closed under *modus ponens*; that is, if $\boldsymbol{A}$, $\boldsymbol{A} \to \boldsymbol{B} \in L$ then $\boldsymbol{B} \in L$,

(iii) L is closed under substitution; that is, if $\boldsymbol{A}'$ is the result of substituting a formula $\boldsymbol{B}$ for some propositional letter $\boldsymbol{P}$ in $\boldsymbol{A}$, then $\boldsymbol{A} \in L$ implies $\boldsymbol{A}' \in L$.

By a *normal logic* we understand a logic that also satisfies the following two conditions:

(iv) L contains every formula of the form

(#0) $[\alpha](\boldsymbol{A} \to \boldsymbol{B}) \to ([\alpha]\boldsymbol{A} \to [\alpha]\boldsymbol{B})$,

(v) L is closed under α-necessitation, for every α; that is, if $\boldsymbol{A} \in L$ then also $[\alpha]\boldsymbol{A} \in L$.

Finally, by a *program logic* we understand a normal logic that contains all formulas of the following form:

(#1a) $[\alpha+\beta]\boldsymbol{A} \to [\alpha]\boldsymbol{A}$,

(#1b) $[\alpha+\beta]\boldsymbol{A} \to [\beta]\boldsymbol{A}$,

(#1c) $[\alpha]\boldsymbol{A} \to ([\beta]\boldsymbol{A} \to [\alpha+\beta]\boldsymbol{A})$,

(#2a) $[\alpha\beta]\boldsymbol{A} \to [\alpha][\beta]\boldsymbol{A}$,

(#2b) $[\alpha][\beta]\boldsymbol{A} \to [\alpha\beta]\boldsymbol{A}$,

(#3a) $[\alpha^*]\boldsymbol{A} \to \boldsymbol{A}$,

(#3b) $[\alpha^*]\boldsymbol{A} \to [\alpha]\boldsymbol{A}$,

(#3c) $[\alpha^*]\boldsymbol{A} \to [\alpha^*][\alpha^*]\boldsymbol{A}$,

(#3d) $\boldsymbol{A} \to ([\alpha^*](\boldsymbol{A} \to [\alpha]\boldsymbol{A}) \to [\alpha^*]\boldsymbol{A})$.

This way of defining program logics seems quite perspicuous. For example, it makes it at once clear that $[\alpha^*]$ is (at least) an S4-modality. However, it is not the shortest definition possible, particularly if non-primitive notation is used. Thus (## 1a–c) can be replaced by the single schema

$$(\#1)\qquad [\alpha+\beta]\boldsymbol{A}\leftrightarrow[\alpha]\boldsymbol{A}\wedge[\beta]\boldsymbol{A},$$

(## 2a, b) by the single schema

$$(\#2)\qquad [\alpha\beta]\boldsymbol{A}\leftrightarrow[\alpha][\beta]\boldsymbol{A},$$

and (## 3a–c) by the single schema

$$(\#3)\qquad [\alpha^*]\boldsymbol{A}\to\boldsymbol{A}\wedge[\alpha][\alpha^*]\boldsymbol{A}.$$

The most remarkable of the preceding schemata is (# 3d), a kind of induction schema (cf. the counterparts in tense logic due to Dana Scott and E. J. Lemmon that are discussed in Prior [9], pp. 66ff.).

Any modal logic, normal or not, induces a deducibility relation as follows. Let us say that a formula $\boldsymbol{A}$ is *deducible in a logic L from a set Σ of formulas, in symbols* $\Sigma\vdash_L\boldsymbol{A}$, if there is a finite number of formulas $\boldsymbol{B}_0,\ldots,\boldsymbol{B}_{n-1}\in\Sigma$ such that

$$\boldsymbol{B}_0\wedge\ldots\wedge\boldsymbol{B}_{n-1}\to\boldsymbol{A}\in L.$$

If we write $\vdash_L\boldsymbol{A}$ for $\emptyset\vdash_L\boldsymbol{A}$, it follows that $\vdash_L\boldsymbol{A}$ if and only if $\boldsymbol{A}\in L$, that is, the theses of L are exactly the formulas that are deducible in L from the empty set. Notice that L is closed under deducibility. Hence $\boldsymbol{A}\in L$ if and only if $\boldsymbol{A}$ is a thesis of L.

We say that a set Σ of formulas is *L-consistent* if not every formula is deducible in L from Σ. An *L-inconsistent* set is of course one that is not L-consistent. It is readily seen that a set Σ is L-consistent if and only if every finite subset of Σ is L-consistent.

Let P be the smallest program logic (P for Pratt). The following claim is the main result to be established in this paper:

THEOREM 3.1. *P coincides with the set of program logically true formulas.*

As usual in modal logic, the soundness part of this claim is easy to establish: that every formula in P (thesis of P) has the property of being true in every program model is shown by the fact that all tautologies and all instances of (## 0–3) have this property, and that *modus ponens* and α-necessitation preserve it. It is the converse that it is difficult to prove.

4. Canonical models

Suppose that L is a normal logic. By the *canonical model* of L we mean the triple $\mathfrak{M}_L=\langle U_L,R_L,V_L\rangle$ where

(i) U_L is the set of all maximal L-consistent sets of formulas,

(ii) for every program expression α and for all $u,v\in U_L$,

$$uR_L(\alpha)v \text{ iff, for all } \boldsymbol{A}, \text{ if } [\alpha]\boldsymbol{A}\in u \text{ then } \boldsymbol{A}\in v,$$

(iii) for every propositional letter $\boldsymbol{P}$,

$$V_L(\boldsymbol{P})=\{u\in U_L:\ \boldsymbol{P}\in u\}.$$

The following theorem generalizes Scott's well known result in modal logic:

THE TRUTH LEMMA 4.1. *If L is any normal logic, then, for all $u\in U_L$ and for all formulas $\boldsymbol{A}$,*

$$\mathfrak{M}_L\vDash_u\boldsymbol{A}\textit{ if and only if }\boldsymbol{A}\in u.$$

The proof — a straightforward induction on the complexity of $\boldsymbol{A}$ — is omitted.

COROLLARY 4.2. *If L is any normal logic, then $\boldsymbol{A}$ is a thesis of L if and only if $\boldsymbol{A}$ is true in $\mathfrak{M}_L$.*

Proof. If $\vdash_L\boldsymbol{A}$, then $\boldsymbol{A}$ is an element of every maximal L-consistent set and hence, by the Truth Lemma, true in $\mathfrak{M}_L$. On the other hand, if not $\vdash_L\boldsymbol{A}$, then $\{\neg\boldsymbol{A}\}$ is an L-inconsistent set which, by Lindenbaum's Lemma, possesses some maximal L-consistent extension, say x. With $x\in U_L$ and $\boldsymbol{A}\notin x$, the Truth Lemma entails the falsity of $\mathfrak{M}_L\vDash_x\boldsymbol{A}$. ∎

The canonical model exists and the Truth Lemma holds for every normal logic, hence in particular for P, the smallest program logic. It is clear from the corollary, then, that if $\mathfrak{M}_P$ happened to be a program model, the completeness problem for P, which is our concern here, would be solved. As the next few lemmata show, $\mathfrak{M}_P$ is almost a program model — almost but not quite. (Actually, for the completeness proof we only need the $\subseteq$-parts of Lemmata 4.3A and 4.3B.)

LEMMA 4.3A. *If L is a program logic, then, for all α and β,*

$$R_L(\alpha+\beta)=R_L(\alpha)\cup R_L(\beta).$$

Proof. Assume that $uR_L(\alpha)v$. Suppose that $[\alpha+\beta]\boldsymbol{A}\in u$. By (# 1a), $[\alpha]\boldsymbol{A}\in u$. Hence, as $uR_L(\alpha)v$, $\boldsymbol{A}\in v$. This shows that $R_L(\alpha+\beta)\supseteq R_L(\alpha)$. An analogous argument, invoking (# 1b), shows that $R_L(\alpha+\beta)\supseteq R_L(\beta)$.

For the converse, assume that neither $uR_L(\alpha)v$ nor $uR_L(\beta)v$. Then there exist $\boldsymbol{A}$ and $\boldsymbol{B}$ such that

$$[\alpha]\boldsymbol{A}\in u,\qquad \boldsymbol{A}\notin v,$$
$$[\beta]\boldsymbol{B}\in u,\qquad \boldsymbol{B}\notin v.$$

Consequently, $[\alpha](\boldsymbol{A}\vee\boldsymbol{B})\in u$, and $[\beta](\boldsymbol{A}\vee\boldsymbol{B})\in u$ and so, by (# 1c), $[\alpha+\beta](\boldsymbol{A}\vee\boldsymbol{B})\in u$. On the other hand, $\boldsymbol{A}\notin v$ and $\boldsymbol{B}\notin v$ implies that $\boldsymbol{A}\vee\boldsymbol{B}\notin v$. Therefore, not $uR_L(\alpha+\beta)v$. This shows that $R_L(\alpha+\beta)\subseteq R_L(\alpha)\cup R_L(\beta)$. ∎

LEMMA 4.3B. *If L is a program logic, then, for all α and β,*

$$R_L(\alpha\beta) = R_L(\alpha)|R_L(\beta).$$

Proof. The easy half of this proof — the $\supseteq$-part — is similar to the corresponding part of the proof of the preceding lemma; here one uses the fact that L contains all instances of (#2a).

For the other half — the $\subseteq$-part — assume that

(1) $$uR_L(\alpha\beta)v.$$

Let $\boldsymbol{C}_0, \boldsymbol{C}_1, \ldots, \boldsymbol{C}_n, \ldots$ be an exhaustive enumeration of all formulas in v. We define a new sequence of formulas as follows:

$$\boldsymbol{B}_0 = \boldsymbol{C}_0,$$
$$\boldsymbol{B}_{n+1} = \boldsymbol{B}_n \wedge \boldsymbol{C}_{n+1}.$$

Note that

(2) for all n, $\boldsymbol{B}_n \in v$,

(3) for all n and p, $\boldsymbol{B}_{n+p} \vdash_L \boldsymbol{B}_n$.

Consider the set

$$\Delta = \{\boldsymbol{A}\colon [\alpha]\boldsymbol{A} \in u\} \cup \{\neg[\beta]\neg\boldsymbol{B}_n\colon n < \omega\}.$$

We claim that Δ is L-consistent. Suppose not! Then there are formulas $\boldsymbol{A}_0, \ldots, \boldsymbol{A}_{m-1}$ and natural numbers $i_0, \ldots, i_{n-1}$ such that

(4) $$[\alpha]\boldsymbol{A}_0, \ldots, [\alpha]\boldsymbol{A}_{m-1} \in u,$$

(5) $\{\boldsymbol{A}_0, \ldots, \boldsymbol{A}_{m-1}, \neg[\beta]\neg\boldsymbol{B}_{i_0}, \ldots, \neg[\beta]\neg\boldsymbol{B}_{i_{n-1}}\}$ is an L-inconsistent set.

Suppose that $k = \max\{i_0, \ldots, i_{n-1}\}$. Then (3) and (5) imply that the set $\{\boldsymbol{A}_0, \ldots, \boldsymbol{A}_{m-1}, \neg[\beta]\neg\boldsymbol{B}_k\}$ is L-inconsistent. From this it follows that

$$\vdash_L \boldsymbol{A}_0 \wedge \ldots \wedge \boldsymbol{A}_{m-1} \to [\beta]\neg\boldsymbol{B}_k.$$

Using the fact that L is normal, we conclude (applying α-necessitation, (#0) and truth-functional reasoning) that

$$\vdash_L [\alpha]\boldsymbol{A}_0 \wedge \ldots \wedge [\alpha]\boldsymbol{A}_{m-1} \to [\alpha][\beta]\neg\boldsymbol{B}_k.$$

This, in conjunction with (4), yields $[\alpha][\beta]\neg\boldsymbol{B}_k \in u$. Hence, by (#2b), $[\alpha\beta]\neg\boldsymbol{B}_k \in u$. By (1), then, $\neg\boldsymbol{B}_k \in v$. Therefore, since v is L-consistent, $\boldsymbol{B}_k \notin v$ which contradicts (2).

The L-consistency of Δ, thus established, enables us to conclude, by Lindenbaum's Lemma, that there is some $x \in U_L$ such that $\Delta \subseteq x$. It is easy to show — this is of course why Δ was defined the way it was! — that $uR_L(\alpha)x$ and $xR_L(\beta)v$. ■

LEMMA 4.3C. *If L is a program logic, then*

$$R_L(\alpha^*) \supseteq (R_L(\alpha))^*.$$

Proof. Easy, since L contains all instances of (##3a–c). ■

If the converse of Lemma 4.3C held, the canonical model of a program logic would be a program model. But is does not! To see that it does not hold for P, let $\boldsymbol{P}$ be any propositional letter and π any program letter. Consider the set

$$\Gamma = \{[\pi]^n\boldsymbol{P}\colon n < \omega\} \cup \{\neg[\pi^*]\boldsymbol{P}\}.$$

For every n, let $\mathfrak{A}_n = \langle N, R, V_n\rangle$ be some program model such that N is the set of natural numbers, $R(\pi)$ is the relation of immediate successor, and

$$V_n(\boldsymbol{P}) = \{i\colon i \leqslant n\}.$$

(Thus it is only the valuation that depends on n.) If Γ' is any finite subset of Γ, then there will be some j such that $[\pi]^j\boldsymbol{P} \in \Gamma'$, while, for all k, $[\pi]^{j+k}\boldsymbol{P} \notin \Gamma'$. It is easy to see that $\mathfrak{A}_j$ is a model for Γ' in the sense that, for all $\boldsymbol{A} \in \Gamma'$, $\mathfrak{A}_j \vDash_0 \boldsymbol{A}$. Hence Γ' is P-consistent. But if every finite subset of Γ is P-consistent, then Γ itself is P-consistent. Hence, again by Lindenbaum's Lemma, Γ can be extended to some $x \in U_L$. By the Truth Lemma we conclude, *pro primo*, that there is some $y \in U_L$ such that $xR_L(\pi^*)y$ and $\boldsymbol{P} \notin y$ (since $\neg[\pi^*]\boldsymbol{P} \in x$), and, *pro secundo*, that $x(R_L(\pi))^*y$ does not hold (since $[\pi]^n\boldsymbol{P} \in x$ implies that $x(R_L(\pi))^n z$ only if $\boldsymbol{P} \in z$). *Ergo*, the converse of Lemma 4.3C fails.

5. Filtrations

By the set of *subformulas* of a given formula $\boldsymbol{A}$ we understand the smallest set Σ such that the following conditions are satisfied:

(i) $\boldsymbol{A} \in \Sigma$,

(ii) if $\neg\boldsymbol{B} \in \Sigma$, then $\boldsymbol{B} \in \Sigma$,

(iii) if $\boldsymbol{B} \to \boldsymbol{C} \in \Sigma$, then $\boldsymbol{B}, \boldsymbol{C} \in \Sigma$,

(iv) if $[\alpha]\boldsymbol{B} \in \Sigma$, then $\boldsymbol{B} \in \Sigma$.

Let L be any normal logic. Any set Ψ of formulas induces an equivalence relation on U_L as follows:

$$u \equiv v \text{ if and only if } u \cap \Psi = v \cap \Psi.$$

We write $|u|$ for the equivalence class of u. Note that $\equiv$ and hence $|u|$ expressly depend on Ψ, even though our symbolism does not reflect the dependency.

Let Ψ be any set closed under subformulas (that is, whenever $\boldsymbol{A}$ is a subformula of $\boldsymbol{B}$, then $\boldsymbol{B} \in \Psi$ only if $\boldsymbol{A} \in \Psi$). We say that a model

$\mathfrak{M} = \langle \bar{U}, \bar{R}, \bar{V} \rangle$ is a *filtration of* the canonical model $\mathfrak{M}_L$ for L *through* Ψ if the following conditions obtain:

(i) $\bar{U} = \{|u|: u \in U_L\}$,

(ii) for every program expression α in Ψ, if $u\, R_L(\alpha)\, v$, then

$$|u|\, \bar{R}(\alpha)\, |v|,$$

(iii) for every program expression α in Ψ, if $|u|\, \bar{R}(\alpha)\, |v|$, then, for all $\boldsymbol{A}$,

$$[\alpha]\boldsymbol{A} \in u \cap \Psi \text{ only if } \boldsymbol{A} \in v,$$

(iv) for every propositional letter $\boldsymbol{P}$ in Ψ,

$$\bar{V}(\boldsymbol{P}) = \{|u|: u \in V_L(\boldsymbol{P})\}.$$

The following is a generalization of a well-known result in modal logic:

THE FILTRATION THEOREM 5.1. *For all $u \in U_L$ and all formulas $\boldsymbol{A} \in \Psi$,*

$$\bar{\mathfrak{M}} \models_{|u|} \boldsymbol{A} \textit{ if and only if } \boldsymbol{A} \in u.$$

We omit the proof as it is a straightforward induction on the complexity of $\boldsymbol{A}$. Notice, however, that it is vital that Ψ is closed under subformulas. It may also be noticed that the theorem can be stated in a slightly stronger form: it holds for any $\boldsymbol{A}$ that is a Boolean combination of formulas in Ψ.

In the statement of the Filtration Theorem nothing has been assumed about the cardinality of Ψ. It is worth noting, though, that if Ψ is finite, then so is $\bar{U}$. In fact, if card$\Psi = n$, then card $\bar{U} \leqslant 2^n$.

6. The completeness proof

By the *Fischer–Ladner closure* of a set Ψ of formulas we understand the smallest set Σ that satisfies the following conditions:

(i) $\Psi \subseteq \Sigma$,

(ii) Σ is closed under subformulas,

(iii) if $[\alpha+\beta]\boldsymbol{A} \in \Sigma$, then $[\alpha]\boldsymbol{A}$, $[\beta]\boldsymbol{A} \in \Sigma$,

(iv) if $[\alpha\beta]\boldsymbol{A} \in \Sigma$, then $[\alpha][\beta]\boldsymbol{A} \in \Sigma$,

(v) if $[\alpha^*]\boldsymbol{A} \in \Sigma$, then $[\alpha][\alpha^*]\boldsymbol{A} \in \Sigma$.

(Conditions (iii)–(v) may be compared to (## 1–3).) We say that a set is *closed under the Fischer–Ladner conditions* if it is its own Fischer–Ladner closure. The following result is proved in Fischer and Ladner [2]:

THE FISCHER–LADNER LEMMA 6.1. *The Fischer–Ladner closure of a finite set is finite.*

We come now to the main observation of this paper. Let Ψ be a formula set closed under subformulas. Let $\mathfrak{M}^\dagger = \langle \bar{U}, R^\dagger, \bar{V} \rangle$ be any program model such that, for all program letters π in Ψ,

$$|u|\, R^\dagger(\pi)\, |v| \quad \text{iff} \quad \exists u_0 \equiv u \exists v_0 \equiv v \big(u_0\, R_L(\pi)\, v_0\big).$$

Such models certainly exist. (There are usually more than one since there are no conditions on $R^\dagger(\pi)$ if π is a program letter not in Ψ, and also no conditions on $\bar{V}(\boldsymbol{P})$ if $\boldsymbol{P}$ is a propositional letter not in Ψ. Actually, $\mathfrak{M}^\dagger$ is a special case of a construction in [2] and a generalization of one in [10].)

THEOREM 6.2. *Suppose that L is a program logic and that Ψ is a finite set of formulas closed under the Fischer–Ladner conditions. Then $\mathfrak{M}^\dagger$ is a filtration of $\mathfrak{M}_L$ through Ψ.*

The proof of the theorem reduces to proving two lemmata, viz., that $R^\dagger$ satisfies conditions (ii) and (iii) in the definition of filtration. In the statement of those lemmata we do not explicitly repeat the three vital assumptions of the theorem: that L is a program logic, that Ψ is closed under the Fischer–Ladner conditions, and that Ψ is finite. But they are all needed!

LEMMA 6.3A. *For all α in Ψ, if $u\, R_L(\alpha)\, v$, then $|u|\, R^\dagger(\alpha)\, |v|$.*

Proof. By induction on α. The basic step follows from the definition of $R^\dagger$. The inductive step consists of three parts.

(I) Suppose that the lemma holds for α and β, and that $u\, R_L(\alpha+\beta)\, v$. By Lemma 4.3A, $u\, R_L(\alpha)\, v$ or $u\, R_L(\beta)\, v$, so, by the induction hypothesis, $|u|\, R^\dagger(\alpha)\, |v|$ or $|u|\, R^\dagger(\beta)\, |v|$. Since $\mathfrak{M}^\dagger$ is a program model, in either case $|u|\, R^\dagger(\alpha+\beta)\, |v|$.

(II) Suppose that the lemma holds for α and β, and that $u\, R_L(\alpha\beta)\, v$. By Lemma 4.3B, there exists some $x \in U_L$ such that $u\, R_L(\alpha)\, x$ and $x\, R_L(\beta) v$. Hence, by the induction hypothesis, $|u|\, R^\dagger(\alpha)\, |x|$ and $|x|\, R^\dagger(\beta)\, |v|$. Since $\mathfrak{M}^\dagger$ is a program model, $|u|\, R^\dagger(\alpha\beta)\, |v|$.

(III) Suppose that the lemma holds for α, and that

$$u\, R_L(\alpha^*)\, v. \tag{1}$$

This is the difficult part of the proof; in fact, the difficulty encountered here is the heart of the whole matter. It is now that we shall finally make use of the induction schema (# 3d), which has not been used so far. One may say that the difficulty in constructing a completeness proof consists in manoeuvring oneself into a position in which one is able to tap the power of this schema.

Suppose, by way of contradiction, that

(2) it is not the case that $|u|\ R^{\dagger}(\alpha^*)\ |v|$.

It is an important fact — see [11], pp. 31f. — that, since $\overline{U}$ is finite, it is possible to find a Boolean combination B of formulas in Ψ such that

(3) $$\text{for all } w \in U_L,\quad B \in w \quad \text{iff} \quad |u|\ R^{\dagger}(\alpha^*)\ |w|.$$

Since $\mathfrak{M}^{\dagger}$ is a program model, $R^{\dagger}(\alpha^*)$ is reflexive. Hence, by (3),

(4) $$B \in u.$$

Moreover, by (2) and (3), $B \notin v$. Hence, by (1),

(5) $$[\alpha^*]B \notin u.$$

Since L is a program logic we may now appeal to (#3d) to infer, from (4) and (5), that

$$[\alpha^*](\boldsymbol{B} \to [\alpha]\boldsymbol{B}) \notin u.$$

Consequently there exist $x, y \in U_L$ such that $u\ R_L(\alpha^*)\ x$ and

(6) $$B \in x,$$

(7) $$x\ R_L(\alpha)\ y,$$

(8) $$B \notin y.$$

By (3) and (6),

(9) $$|u|\ R^{\dagger}(\alpha^*)\ |x|.$$

By the induction hypothesis, (7) implies that

(10) $$|x|\ R^{\dagger}(\alpha)\ |y|.$$

Since $\mathfrak{M}^{\dagger}$ is a program model, (9) and (10) imply that

(11) $$|u|\ R^{\dagger}(\alpha^*)\ |y|.$$

But, according to (3) and (11), $B \in y$, which is impossible in view of (8). ∎

LEMMA 6.3B. *For all a in Ψ, if $|u|\ R^{\dagger}(\alpha)\ |v|$, then, for all $\boldsymbol{A}$, $[\alpha]\boldsymbol{A} \in u \cap \Psi$ only if $\boldsymbol{A} \in v$.*

Proof. By induction on α. The basic step follows readily from the definition of $R^{\dagger}$. The inductive step again consists of three parts.

(I) Suppose that the lemma holds for α and β, and that $|u|\ R^{\dagger}(\alpha+\beta)\ |v|$. Take any $\boldsymbol{A}$ such that $[\alpha+\beta]\boldsymbol{A} \in u \cap \Psi$. Hence, by (##1a,b) on one hand and the fact that Ψ is closed under the Fischer–Ladner conditions on the other, it follows that $[\alpha]\boldsymbol{A}$, $[\beta]\boldsymbol{A} \in u \cap \Psi$. Since $\mathfrak{M}^{\dagger}$ is a program model, either $|u|\ R^{\dagger}(\alpha)\ |v|$ or $|u|\ R^{\dagger}(\beta)\ |v|$. So, in either case, the induction hypothesis gives us $\boldsymbol{A} \in v$.

(II) Suppose that the lemma holds for α and β, and that $|u|\ R^{\dagger}(\alpha\beta)\ |v|$. Take any $\boldsymbol{A}$ such that $[\alpha\beta]\boldsymbol{A} \in u \cap \Psi$. Then, by (#2a) and the fact that Ψ is closed under the Fischer–Ladner conditions, $[\alpha][\beta]\boldsymbol{A} \in u \cap \Psi$. Since $\mathfrak{M}^{\dagger}$ is a program model, there exists some $x \in U_L$ such that $|u|\ R^{\dagger}(\alpha)\ |x|$ and $|x|\ R^{\dagger}(\beta)\ |v|$. Therefore, by the induction hypothesis, $[\beta]\boldsymbol{A} \in x$. Hence, by another appeal to the Fischer–Ladner conditions and the induction hypothesis, $\boldsymbol{A} \in v$.

(III) Suppose that the lemma holds for α, and that $|u|\ R^{\dagger}(\alpha^*)\ |v|$. This case is just slightly more involved than the others. Take any $\boldsymbol{A}$ such that $[\alpha^*]\boldsymbol{A} \in u \cap \Psi$. We claim that, for all $x, y \in U_L$,

(§) $$\text{for all } i, \text{ if } |x|\ \big(R^{\dagger}(\alpha)\big)^i\ |y|, \text{ then } [\alpha^*]\boldsymbol{A} \in x \text{ only if } [\alpha^*]\boldsymbol{A} \in y.$$

The claim is proved by induction on i. The case $i = 0$ is trivial. Suppose that the claim holds for n, and that

(1) $$|x|\ \big(R^{\dagger}(\alpha)\big)^{n+1}\ |y|.$$

Assume that $[\alpha^*]\boldsymbol{A} \in x$. Recall that, in the presence of (##3b, c), $\vdash_L [\alpha^*]\boldsymbol{A} \to [\alpha][\alpha^*]\boldsymbol{A}$. For this reason, and since Ψ is closed under the Fischer–Ladner conditions,

(2) $$[\alpha][\alpha^*]\boldsymbol{A} \in x \cap \Psi.$$

By (1) there is some $z \in U_L$ such that

(3) $$|x|\ R^{\dagger}(\alpha)\ |z|,$$

(4) $$|z|\ \big(R^{\dagger}(\alpha)\big)^{n}\ |y|.$$

From (2) and (3) it follows, by the induction hypothesis on α, that $[\alpha^*]\boldsymbol{A} \in z$. Hence, by (4) and the induction hypothesis on i, $[\alpha^*]\boldsymbol{A} \in y$. This ends the proof of (§).

Now $\mathfrak{M}^{\dagger}$ is a program model. Hence our assumption that $|u|\ R^{\dagger}(\alpha^*)\ |v|$ implies that, for some j, $|u|\ \big(R^{\dagger}(\alpha)\big)^j |v|$. Therefore, by (§), $[\alpha^*]\boldsymbol{A} \in v$. Hence, by (#3a), $\boldsymbol{A} \in v$. ∎

It should be clear that we have now reached our goal. For suppose that $\boldsymbol{A}$ is any non-thesis of P. Then, by Corollary 4.2, there is some $x \in U_L$ such that $\boldsymbol{A} \notin x$. By Lemma 6.1, the Fischer–Ladner closure Ψ of the set $\{\boldsymbol{A}\}$ is finite. Let $\mathfrak{M}^{\dagger}$ be as described. By Theorem 6.2, $\mathfrak{M}^{\dagger}$ is a filtration, and by the Filtration Theorem 5.1, then, $\boldsymbol{A}$ fails to be true at $|x|$ in $\mathfrak{M}^{\dagger}$. And $\mathfrak{M}^{\dagger}$ is a program model by definition!

7. The inverse operator

In this section we shall show that the preceding discussion can easily be modified to accommodate also another program operator of importance, viz., the inverse operator.

Thus let $^{-1}$ be added as a new unary program operator. In keeping with our other conventions we shall write α^{-1} rather than $^{-1}(\alpha)$. In the definition of *program model* we add the condition that, for all α,

(iv) $$R(\alpha^{-1}) = \breve{R}(\alpha),$$

where $\breve{}$ represents the converse; that is, if S is any binary relation, then

$$\breve{S} = \{\langle x, y\rangle : \langle y, x\rangle \in S\}.$$

The notion of a *program logic* is extended by requiring inclusion of all instances of the schemata

(# 4a) $\neg[\alpha]\neg[\alpha^{-1}]\boldsymbol{A} \to \boldsymbol{A}$,

(# 4b) $\neg[\alpha^{-1}]\neg[\alpha]\boldsymbol{A} \to \boldsymbol{A}$.

The definition of canonical model is not affected by these changes, but there is a new observation to add to Lemmata 4.3A–C (as Lemma 4.3D as it were):

LEMMA 7.1. *If L is a program logic, then for all α,*

$$R_L(\alpha^{-1}) = \breve{R}_L(\alpha).$$

Proof. Suppose that $uR_L(\alpha^{-1})v$. Take any $\boldsymbol{A}$ such that $[\alpha]\boldsymbol{A} \in v$. Then it is impossible that $[\alpha^{-1}]\neg[\alpha]\boldsymbol{A} \in u$, so $\neg[\alpha^{-1}]\neg[\alpha]\boldsymbol{A} \in u$. Hence, by (# 4b), $\boldsymbol{A} \in u$. Therefore $vR_L(\alpha)u$, and so $u\breve{R}_L(\alpha)v$.

Conversely, suppose that $u\breve{R}_L(\alpha)v$. Then $vR_L(\alpha)u$. Take any $\boldsymbol{A}$ such that $[\alpha^{-1}]\boldsymbol{A} \in u$. Then evidently $\neg[\alpha]\neg[\alpha^{-1}]\boldsymbol{A} \in v$. Hence, by (# 4a), $\boldsymbol{A} \in v$. So $uR(\alpha^{-1})v$. ■

The proofs of the Truth Lemma 4.1 and the Filtration Theorem 5.1 go through as before. The definition of *Fischer–Ladner closure* is extended by the new condition

(vi) if $[\alpha^{-1}]\boldsymbol{A} \in \Sigma$, then $[\alpha]\neg[\alpha^{-1}]\boldsymbol{A} \in \Sigma$.

It now becomes necessary to prove that the Fischer–Ladner Lemma 6.1 continues to hold in this more general setting. It does, but we omit the proof (which offers no new difficulty).

The definition of $\mathfrak{M}^\dagger$ is taken over word for word, and Theorem 6.2 is stated as before, but the terms "program logic", "Fischer–Ladner conditions", etc., are of course now understood in the new, inclusive

sense. The proofs of Lemmata 6.3A, B need the following amendments to their respective inductive steps:

Ad Lemma 6.3A: (IV) Suppose that the lemma holds for α, and that $u\,R_L(\alpha^{-1})\,v$. By Lemma 7.1, $v\,R_L(\alpha)\,u$. Hence, by the induction hypothesis, $|v|\,R^\dagger(\alpha)\,|u|$. Since, by definition, $\mathfrak{M}^\dagger$ is a program model, $|u|\,R^\dagger(\alpha^{-1})\,|v|$. ■

Ad Lemma 6.3B: (IV) Suppose that the lemma holds for α, and that $|u|\,R^\dagger(\alpha^{-1})\,|v|$. Take any $\boldsymbol{A}$ such that $[\alpha^{-1}]\boldsymbol{A} \in u \cap \Psi$. Note that by the new Fischer–Ladner condition, $[\alpha]\neg[\alpha^{-1}]\boldsymbol{A} \in \Psi$. Since $\mathfrak{M}^\dagger$ is a program model, $|v|\,R^\dagger(\alpha)\,|u|$. Therefore, if $[\alpha]\neg[\alpha^{-1}]\boldsymbol{A} \in v$, it would follow by the induction hypothesis that $\neg[\alpha^{-1}]\boldsymbol{A} \in u$, contradicting the consistency of u. Consequently $\neg[\alpha]\neg[\alpha^{-1}]\boldsymbol{A} \in v$, and so, by (#4a), $\boldsymbol{A} \in v$. ■

With these modifications we have a new completeness result. Thus, like $+$ and $\cdot$, $^{-1}$ is a very well-behaved operator, from our point of view.

8. Remarks on the background of this paper

The completeness problem solved in Section 6 was put by Richard Ladner to the participants of a small workshop arranged at Simon Fraser University by S. K. Thomason in early 1977. The author, who was present, became interested when he realized that the main difficulty of the problem — now isolated as part (III) of the inductive step in the proof of Lemma 6.3A — was the same as a problem he had encountered and left open in [12].

A solution along the present lines was developed during the summer of 1977 while the author was a visitor in the philosophy department at the University of Calgary. On the basis of that work an abstract [13] was prepared. The solution itself was presented on July 25, 1977, in a seminar given jointly by Brian F. Chellas and the author.

In early 1978, however, the author discovered a gap in the putative proof. In the meanwhile other proofs had been found, independently, by other researchers: Rohit Parikh, Vaughan Pratt, and perhaps others. Thus, even though the claims made in [13] are correct, and the proof in this paper is his, the author cannot claim to have produced the first correct proof. This distinction would seem to belong to Parikh, whose proof — which also covers the inverse operator discussed in Section 7 — appears in [5]. Pratt's proof is sketched in [7].

References

[1] D. M. Gabbay, *Investigations in modal and tense logics with applications to problems in philosophy and linguistics*, Reidel, Dordrecht 1976.

[2] Michael J. Fischer and Richard E. Ladner, *Propositional modal logic of programs*,

Extended abstract, presented at the Ninth A. C. M. Symposium on Theory of Computing, Boulder, Colorado, May 2–4, 1977.

[3] G. E. Hughes and M. J. Cresswell, *Introduction to modal logic*, Methuen, London 1968.

[4] E. J. Lemmon, in collaboration with Dana Scott, *Introduction to modal logic*, *American Philosophical Quarterly*, monograph series, vol. 11, Blackwell, Oxford 1977.

[5] Rohit Parikh, *A completeness result for a propositional dynamic logic*, Forthcoming.

[6] V. R. Pratt, *Semantical considerations on Floyd–Hoare logic*, in: *17th I. E. E. E. Symposium on Foundations of Computer Science* (1977), 109–121.

[7] —, *The tableau method for propositional dynamic logic*, Manuscript.

[8] —, *Logic of processes*, Manuscript.

[9] Arthur Prior, *Past, present and future*, Clarendon Press, Oxford 1967.

[10] Krister Segerberg, *Decidability of S4.1*, Theoria 34 (1968), 7–20.

[11] —, *An essay in classical modal logic*, Uppsala: The Philosophical Society and the Department of Philosophy, University of Uppsala, 1971.

[12] —, *von Wright's tense logic*, Completed in 1974 for *The philosophy of Georg Henrik von Wright*, to be edited by P. A. Schilpp.

[13] —, *A completeness theorem in the modal logic of programs*, Notices of the American Mathematical Society 24 (1977), A–552, no. 77T–E69.

Presented to the Semester
Universal Algebra and Applications
(February 15 – June 9, 1978)

UNIVERSAL ALGEBRA AND APPLICATIONS
BANACH CENTER PUBLICATIONS, VOLUME 9
PWN—POLISH SCIENTIFIC PUBLISHERS
WARSAW 1982

BROUWERIAN SEMILATTICES: THE LATTICE OF TOTAL SUBALGEBRAS

PETER KÖHLER

Institute of Mathematics, Justus Liebig University, Giessen, F.R.G.

Any Brouwerian semilattice S can be viewed as a (meet-) semilattice acting on itself, the action being relative pseudo-complementation. This may be formalized by considering S as a (universal) algebra with one binary operation (meet) and for every $a \in S$ a unary operation. The subalgebra lattice of this algebra is the main topic of this paper: It is shown that it is a maximal distributive sublattice of the subalgebra lattice of S considered as an (ordinary) Brouwerian semilattice; Brouwerian semilattices are characterized for which this lattice is Boolean. The question which distributive algebraic lattices can be represented this way is left as an open problem.

1. Preliminaries

A *Brouwerian semilattice* is an algebra $\langle S, \wedge, *, 1\rangle$, where $\langle S, \wedge, 1\rangle$ is a meet-semilattice with the greatest element 1, and where the binary operation $*$ is relative pseudocomplementation, i.e. $z \leqslant x*y$ holds for elements $x, y, z \in S$ if and only if $z \wedge x \leqslant y$. Following the usual practice we will mostly identify the Brouwerian semilattice $\langle S, \wedge, *, 1\rangle$ with the underlying set S.

For the basic arithmetic of Brouwerian semilattices we refer to [4], [7]. Let us recall the following rules of computation:

For all $x, y, z \in S$:

(1) $x \leqslant y \Leftrightarrow x*y = 1,$

(2) $1*x = x,$

(3) $x*y \geqslant y,$

(4) $x \wedge x*y = x \wedge y,$

(5) $(x \wedge y)*z = x*(y*z),$