Is propositional calculus categorical?

Jaroslav Peregrin
Department of Logic, Institute of Philosophy, Academy of Sciences of the Czech Republic
www.cuni.cz./~peregrin

Can axioms confer meanings on classical logical connectives?

According to the standard definition, a first-order theory is *categorical* if all its models are isomorphic. The idea behind this definition obviously is that of capturing semantic notions in axiomatic terms: to be categorical is to be, in this respect, successful. Thus, for example, we may want to axiomatically delimit the concept of natural number, as it is given by the pre-theoretic semantic intuitions and reconstructed by the standard model. The well-known results state that this cannot be done within first-order logic, but it can be done within second-order one.

Now let us consider the following question: can we axiomatically capture the semantic concept of conjunction? Such question, to be sure, does not make sense within the standard framework: we cannot construe it as asking whether we can form a first-order (or, for that matter, whatever-order) theory with an extralogical binary propositional operator so that its only model (up to isomorphism) maps the operator on the intended binary truth-function. The obvious reason is that the framework of standard logic does not allow for extralogical constants of this type. But of course there is also a deeper reason: an existence of a constant with this semantics is presupposed by the very definition of the framework[1]. Hence the question about the axiomatic capturability of concunction, if we can make sense of it at all, cannot be asked within the framework of standard logic, we would have to go to a more abstract level. To be able to make sense of the question we would have to think about a propositional 'proto-language', with uninterpreted logical constants, and to try to search out axioms which would fix the denotations of the constants as the intended truth-functions. Can we do this?

It might seem that the answer to this question is yielded by the completeness theorem for the standard propositional calculus: this theorem states that the axiomatic delimitation of the calculus and the semantic delimitation converge to the same result. Hence, it seems, we can make do with the axiom system alone, and hence the axiom system is capable of confering the right meanings on the logical operators. But this is wrong. As is well known, there are theories which do justice to all the axioms and inference rules of the standard logic, which are nevertheless incompatible with the standard truth-functional interpretation of the operators: e.g. theories containing disjunctions together with the negations of both disjuncts[2].

To clarify the situation, we have to delimit the concepts and the framework of out investigation with some rigor. We will take language to be simply a set of sentences with a delimited set of acceptable truth-valuations of the sentences. (We refrain from the discussion of whether this definition is too broad – form the viewpoint of the present paper it is not important[3]). Then we can see an axiomatic system as a way of delimiting the space of acceptable truth-valuations: an axiom is a

[1] We can, of course, switch to another logic, say the intuitionist one, and have a different kind of negation, but then again the semantics of the negation-sign will be fixed 'a priori' (i.e. prior to the framweork being put to use) rather than being up for grabs.

[2] The first one to note it was probably Carnap (1943). See Koslow (1992, Chapter 19) for more details.

[3] I have discussed this questions in detail elsewhere (Peregrin, 1997), where I argued that any kind of semantics *should* be seen as a tool of such a delimitation

sentence which must be be mapped on **1** by every acceptable valuation, whereas a rule gives a sentence which must be mapped on **1** by every valuation which maps some other sentences on **1**. The general question now is which kinds of spaces of acceptable valuations are delimitable in this way, i.e. in terms of axiomatic systems; and in particular whether we can carve the space in such a way that some connectives get pinned down to the standard truth-functions.

Let us be a little bit more precise. Let L be a set and $\boldsymbol{O}$ a function mapping $L\times L$ on L. Let C be a finite set of 'constraints' on valuations of the elements of L, which are of the shape

[if $A_1,\ldots,A_n$ (are true), then also] A (is true).

Where v is a function from L to $\boldsymbol{B}=\{\mathbf{0},\mathbf{1}\}$, we will say that *$v$ respects C* if it does justice to all the constraints of C. Where F is a function from $\boldsymbol{B}\times\boldsymbol{B}$ to $\boldsymbol{B}$, we will say that *C sets the denotation of $\boldsymbol{O}$ to F* the following two conditions are equivalent

(i) v respects C;
(ii) $v(\boldsymbol{O}(A_{1,} A_2)) = \mathrm{F}(v(A_1),v(A_2))$ for every $A_1, A_2\in L$.

Now let $\mathrm{F}_\neg$, $\mathrm{F}_\wedge$, $\mathrm{F}_\vee$ and $\mathrm{F}_\rightarrow$ be the truth-functions standardly assigned to the classical negation, conjunction, disjunction and implication, respectively. Then it is obviously the case that the three constraints

if $A_{1,} A_2$, then $\boldsymbol{O}(A_{1,} A_2)$;
if $\boldsymbol{O}(A_{1,} A_2)$, then A_1; and
if $\boldsymbol{O}(A_{1,} A_2)$, then A_2

set the value of $\boldsymbol{O}$ to $\mathrm{F}_\wedge$. On the other hand, there is no finite set of constraints which would set the value of $\boldsymbol{O}$ to $\mathrm{F}_\neg$, $\mathrm{F}_\vee$, or $\mathrm{F}_\rightarrow$. To see this, consider, for instance, the case of $\mathrm{F}_\vee$. We may clearly help oursevles to the constraints

if A_1, then $\boldsymbol{O}(A_{1,} A_2)$; and
if A_2, then $\boldsymbol{O}(A_{1,} A_2)$.

By them we exclude all valuations which map either A_1 or A_2 on **1**, and $\boldsymbol{O}(A_{1,} A_2)$ on **0**; but what we need in addition to this is exclude also all those which map both A_1 and A_2 on **0** and $\boldsymbol{O}(A_{1,} A_2)$ on **1**. Suppose, for the sake of simplicity, that the language we are considering has no other sentences than A_1, A_2 and $\boldsymbol{O}(A_{1,} A_2)$ (hence that now there is only a single valuation left to be excluded). Then the exhaustive listing of all possible nontrivial constraints, clearly, is the following:

1. A_1
2. A_2
3. $\boldsymbol{O}(A_{1,} A_2)$
4. if A_2, then A_1
5. if $\boldsymbol{O}(A_{1,} A_2)$, then A_1
6. if A_2, $\boldsymbol{O}(A_{1,} A_2)$ then A_1

7. if A_1, then A_2
8. if $\boldsymbol{O}(A_{1,} A_2)$, then A_2
9. if A_1, $\boldsymbol{O}(A_{1,} A_2)$ then A_2
10. if A_1, then $\boldsymbol{O}(A_{1,} A_2)$
11. if A_2, then $\boldsymbol{O}(A_{1,} A_2)$
12. if A_1,A_2, then $\boldsymbol{O}(A_{1,} A_2)$

Out of these, 3., 4., 6., 7., 9., 10., 11., 12. do not exclude the unwanted valuation at all, whereas the other do exclude it, but only at the cost of excluding also some of those valuations which should remain unexcluded.

Of course this is not very surprising: the only thing our constraints can stipulate is that if some sentences are true, also some other sentences are true; but not, e.g., that if some sentences are *false*, other sentences are *false*. But it is only constraints of this very kind which are available to a builder of an axiomatic system. This indicates that many possible spaces of acceptable valuations (and especially some truth-functions) are not categorically delimitable in the axiomatic way.

How does this square with the soundness and completness of the classical, truth-functional logic? To see it, let us first give some more definitions. Let V be a class of mappings of L on $\boldsymbol{B}$. We define

$$\mathrm{Pos}(V) = \{A \in L \mid v(A) = \mathbf{1} \text{ for every } v \in V\}$$
$$\mathrm{Neg}(V) = \{A \in L \mid v(A) = \mathbf{0} \text{ for every } v \in V\}$$

Two classes V_1 and V_2 of valuations are called *positively equivalent* iff $\mathrm{Pos}(V_1) = \mathrm{Pos}(V_2)$; they are called *negatively equivalent* iff $\mathrm{Neg}(V_1) = \mathrm{Neg}(V_2)$.

Let L be a language such that for every its sentence A there is a sentence $\neg A$ and for every two sentences A_1 and A_2 there is a sentence $A_1 \wedge A_2$. We will call its valuation v *classical* iff

$$v(\neg A) = \mathrm{F}_{\neg}(v(A))$$
$$v(A_1 \wedge A_2) = \mathrm{F}_{\wedge}(v(A_1), v(A_2))$$

We will call it *noncontradictory* if for no A it is the case that $v(\neg A)=v(A)=1$; and we will call it *full* iff for no A it is the case that $v(\neg A)=v(A)=0$. (Hence every classical valuation is both noncontradictory and full.)

Calling now the valuation *quasi-classical* iff it respects the axiom system of the classical propositional claculus, what we are going to prove in this paper is the following:

(1) The class of all quasi-clasical valuations is positively equivalent to the class of all classical valuations.

(2) The class of all quasi-clasical valuations *which are noncontradictory* is positively and negatively equivalent to the class of all classical valuations.

(3) The class of all quasi-clasical valuations *which are full and noncontradictory* conicides with the class of all classical valuations.

It is clear that (1) is nothing else than the standard completeness theorem for classical propositional logic (and hence proving merely it would be no achievement). From the current perspective, however, it is only one of the series of results which characterize the relationships between *the class of all quasi-clasical valuations* and *the class of all classical valuations*, other ones being (2) and (3).

Can these results be obtained by a straightforward generalization of the classical completness proof? I do not think so. How do we usually prove (1)? The idea of the most common proof, which Mendelson (1964) attributes to Kalmár (1936), can be construed as follows. For a truth table, let us call a formula (X_1→(...(X_n→X)...)) the *internalization of its i-th row of the table* iff the following holds: (i) if the value in the i-th row and j-th column is 1, then X_i is A_i, whereas in the opposite case it is $\neg A_i$; (ii) if the value in the i-th row and the value column is 1, then X is A, whereas in the opposite case it is $\neg A$. Now Kalmár's proof can be seen as showing that (i) the internalization of any row of the truth table of any formula is a theorem; and (ii) if a formula is a tautology, then it is derivable from the internalizations of the rows of its truth table. As a consequence, a tautology is derivable from theorems, and hence is itself a theorem.

There does not seem to be a way of generalizing this proof to cover not only (1), but also (2) and (3). And what we are going to do within the rest of the paper is to develop a more general framework, within which we can not only prove the theorems sketeched above, but also reach a helpful vantage point to oversee an aspect of the landscape of elementary logic.

Constraints and their internalization

Hence the problem, as we have articulated it in the previous section, is to try to find an axiomatic delimitation of the space of acceptable valuations which would pin down the denotations of some operators to the usual truth-functions. More generally, the problem is that of finding an axiomatic delimitation of a space of valuation delimited in some more general way. Before we turn directly to this task, we give some definitions and prove some simple preparatory results.

DEFINITION 1. A *language* is a set (whose elements are called *sentences*) plus a set of its mappings on $\boldsymbol{B}$={**0**,**1**} (called *acceptable valuations*). A sentence S of L is called *verified by a valuation v* iff $v(S)$ = **1; it** is called *falsified by v* iff $v(S)$ = **0**; it is called *valid* iff it is verified by every acceptable valuation and it is called *countervalid* iff it is falsified by every one. A class C of sentences is called to *entail* a sentence S iff S is verified by every acceptable valuation that verifies all sentences of C. If $\{S_1,...,S_n\}$ entails S, we will write

$S_1,...,S_n \models S$.

It is clear that the concept of validity is reducible to that of entailment: a sentence is valid simply iff it is entailed by an empty set. Now we are going to characterize a class of languages for which there is also an inverse reduction.

DEFINITION 2. Let ► be a binary function from the set of sentences of a language L to the same set; we will write S_1►S_2 instead of ►(S_1,S_2)[4]. ► is called *implication* iff for every sentences $S_1,...,S_n$, S of L

[4] Note that we do not see implication as a *sign*, but rather as a *function*. Of course that in a typical case the function will map any pair of sentences on a sentence built out of them with the help of an implication-sign; but we do not care about the syntax. In this sense, our treatment of implication and negation is close to that of Koslow (1992).

(i) $S_1 \blacktriangleright S_2, S_1 \models S_2$.

(ii) if $S_1,...,S_n \models S$ (where $n>0$), then $S_1,...,S_{n-1} \models S_n \blacktriangleright S$,

If such a function exists, then L is said to *have implication*.

THEOREM 1. Let $\blacktriangleright$ be a implication. Then $S_1 \blacktriangleright(...(S_n \blacktriangleright S)...)$ is valid iff $S_1,...,S_n$ entails S.

PROOF: We first prove the direct implication, by induction. Let first n=1. If $S_1 \blacktriangleright S$ is valid, then, according to (i) of the definition of implication, S must be true whenever S_1 is. Hence if $S_1 \blacktriangleright S$ is valid, then S_1 entails S. Let now the theorem hold for $n=m$; we will show that it holds for $n=m+1$. So let $S_1 \blacktriangleright(...(S_{m+1} \blacktriangleright S)...)$ be valid. Then, according to the inductive assumption, $S_1,...,S_m \models S_{m+1} \blacktriangleright S$. However, according to (i) of the definition of implication, $S_{m+1} \blacktriangleright S$, $S_{m+1} \models S$. But as $S_1,...,S_m,S_{m+1}$ entails both $S_{m+1} \blacktriangleright S$ and S_{m+1}, then, due to the obvious transitivity of entailment, $S_1,...,S_m,S_{m+1} \models S$.

Now we prove the inverse implication, again by induction. Let first n=1. If $S_1 \models S$, then $S_1 \blacktriangleright S$ is valid according to (ii) of the definition of implication. Let now the theorem hold for $n=m$; we shall show that it holds for $n=m+1$. Let $S_1,...,S_{m+1}$ entail S. Then, according to (ii) of the definition of implication, $S_1,...,S_m \models S_{m+1} \blacktriangleright S$; and $S_1 \blacktriangleright...(S_{m+1} \blacktriangleright S)...)$ is valid according to the inductive assumption.

The most perspicuous species of implication is constituted by the well-known *material implication*:

DEFINITION 3. A binary function f from the set of sentences of a language L to the same set is called *material implication* iff, for every sentences S_1, S_2 of L, every acceptable valuation verifies $f(S_1,S_2)$ just in case it either falsifies S_1 or verifies S_2.

It is clear that a material implication is an implication; but it can be shown that not every implication is material.

We will need to consider also a relation between sentences which is more general than entailment. We will say that a distribution of truth values among the sentences S_1, ..., S_n *forces* a truth value of a sentence S iff any acceptable valuation which distributes the truth values among S_1, ..., S_n in the former way assigns the latter value to S:

DEFINITION 4. Let L be a language, let S_1, ..., S_n, S be its sentences, and let V_1, ..., V_n, $V \in \{\mathbf{0},\mathbf{1}\}$. We will say that the assignments of V_i to S_i (for i=1 to n) *forces* the assignment of V to S, and will abbreviate this to

$$S_1^{V1}, ..., S_n^{Vn} \models S^V,$$

iff $v(S)=V$ for any acceptable valuation v such that $v(S_i)=V_i$ (for i=1 to n).

An instance of forcing will be called *positive* if the only truth value mentioned in it is **1**, i.e. if it is of the form

$$S_1^{\mathbf{1}}, ..., S_n^{\mathbf{1}} \models S^{\mathbf{1}}.$$

An instance of forcing will be called *absolute* if $n = 0$, i.e. if it is of the form

$$\models S^V.$$

It is clear that every instance of entailment can be seen as a (positive) instance of forcing ($S_1^{\mathbf{1}}, ..., S_n^{\mathbf{1}} \models S^{\mathbf{1}}$ is the same as $S_1, ..., S_n \models S$); hence entailment can be seen as a special case of forcing. Let us now characterize a class of languages for which entailment is *equivalent* with forcing.

DEFINITION 5. An unary function # from sentences to sentences is called *negation* iff no acceptable valuation verifies both S and $\#S$. It is called a *standard* negation if, moreover, every acceptable valuation which does not verify S verifies $\#S$, i.e. if every acceptable valuation verifies S if and only if it falsifies $\#S$.

THEOREM 2. Let # be a negation. Then:

(i) if $S_1^{V1},...,S_n^{Vn} \models \#S_{n+1}^{\mathbf{1}}$, then $S_1^{V1},...,S_n^{Vn} \models S_{n+1}^{\mathbf{0}}$ (in particular, if $\models \#S^{\mathbf{1}}$, then $\models S^{\mathbf{0}}$); and

(ii) if $S_1^{V1},...,S_i^{\mathbf{0}},...,S_n^{Vn} \models S_{n+1}^{Vn+1}$, then $S_1^{V1},...,\#S_i^{\mathbf{1}},...,S_n^{Vn} \models S_{n+1}^{Vn+1}$.

PROOF: (i) Suppose that $S_1^{V1},...,S_n^{Vn} \models \#S_{n+1}^{\mathbf{1}}$. Then $\#S_{n+1}$ is verified by every acceptable valuation which assigns V_i to S_i for i=1 to n. But then every such valuation is bound to falsify S_{n+1}. Hence $S_1^{V1},...,S_n^{Vn} \models S_{n+1}^{\mathbf{0}}$.

(ii) Suppose that $S_1^{V1},...,S_i^{\mathbf{0}},...,S_n^{Vn} \models S_{n+1}^{Vn+1}$. Then every acceptable valuation which assigns V_j to S_j for j=1,...,i-1,i+1,..., n and **0** to S_i assigns V_{n+1} to S_{n+1}. However, as every truth valution which verifies $\#S_i$ is bound to falsify S_i, and hence every acceptable valuation which assigns V_j to S_j for j=1,...,i-1,i+1,..., n, and **1** to $\#S_i$, assigns V_j to S_j for j=1,...,i-1,i+1,..., n, and **0** to S_i. Hence $S_1^{V1},...,\#S_i^{\mathbf{1}},...,S_n^{Vn} \models S_{n+1}^{Vn+1}$.

THEOREM 3. Let # be a standard negation. Then

(i) if $S_1^{V1},...,S_n^{Vn} \models S_{n+1}^{\mathbf{0}}$, then $S_1^{V1},...,S_n^{Vn} \models \#S_{n+1}^{\mathbf{1}}$ (in particular, if $\models S^{\mathbf{0}}$, then $\models \#S^{\mathbf{1}}$); and

(ii) if $S_1^{V1},...,\#S_i^{\mathbf{1}},...,S_n^{Vn} \models S_{n+1}^{Vn+1}$, then $S_1^{V1},...,S_i^{\mathbf{0}},...,S_n^{Vn} \models S_{n+1}^{Vn+1}$.

Hence for every sentences $S_1,...,S_n,S_{n+1}$ and every truth values $V_1,...,V_n,V_{n+1}$ it holds that $S_1^{V1},...,S_n^{Vn}$ force S_{n+1}^{Vn+1} iff $X_1,...,X_n$ entail X_{n+1}, where X_i is S_i if V_i is **1** and it is $\#S_i$ if V_i is **0**. Hence in a language which has a standard negation, any instance of forcing is expressible in the form of an instance of entailment.

PROOF: As a sentence S of L is verified by a acceptable valuation iff $\#S$ is not, the requirement that S is not verified is equivalent to the requirement that #S is.

DEFINITION 6. A language is called *normal* if it has a standard negation and an implication. A language is called *strongly normal* if it has a standard negation and a material implication.

Negation is directly characterized in terms of constraints. The same is not true of implication in general, but it is true about its material version:

THEOREM 4. g is a standard negation iff

(N1) $S^{\mathbf{1}} \models \#S^{\mathbf{0}}$, and

(N2) $S^{\mathbf{0}} \models \#S^{\mathbf{1}}$;

f is a material implication iff

(MI1) $S_1^{\mathbf{0}} \models (S_1 \blacktriangleright S_2)^{\mathbf{1}}$,

(MI2) $S_2^{\mathbf{1}} \models (S_1 \blacktriangleright S_2)^{\mathbf{1}}$, and

(MI3) $S_1^{\mathbf{1}},S_2^{\mathbf{0}} \models (S_1 \blacktriangleright S_2)^{\mathbf{0}}$.

PROOF: Obvious.

Characterizing validity

What we are after can now be generally restated as delimiting the set of acceptable valuations of sentences of a language in terms of positive constraints. If we assume that the set is delimited by (not necessarily positive) constraints (which is the case as far as our main problem is concerned, for the space of acceptable truth valuations of the classical propositional calculus is delimited by the definition of its semantics), our task is that of turning non-positive constraints into positive ones with the smallest possible tampering with their effect. We have seen that if a language has an implication and a standard negation, every its constraint is expressible as positive and absolute – i.e., in effect, as a valid sentence. Let us call a sentence expressing a constraint in this way the *internalization* of the constraint:

DEFINITION 7. Let f and g be a binary resp. a unary function from the set of sentences of a language L to the same set. An (f,g)*-internalization* of a constraint S_1^{V1}, ..., $S_n^{Vn} \models S_{n+1}^{Vn+1}$ is the sentence $f(X_1,...f(X_n,X_{n+1}))$, where X_i is S_i if V_i is $\mathbf{1}$ and is $g(S_i)$ if V_i is $\mathbf{0}$. (Hence the internalization of an absolute constraint $\models S_{n+1}^{Vn+1}$ is X_{n+1}; and that of a positive absolute constraint $\models S_{n+1}^{\mathbf{1}}$ is S_{n+1}.)

THEOREM 5. If ► is an implication and # a standard negation, then S_1^{V1}, ..., S_n^{Vn} forces S_{n+1}^{Vn+1} if and only if the (►,#)-internalization of S_1^{V1}, ..., $S_n^{Vn} \models S_{n+1}^{Vn+1}$ is valid.
PROOF: Due to Theorem 1, S_1^{V1}, ..., $S_n^{Vn} \models S_{n+1}^{Vn+1}$ iff $X_1,...,X_n$ entail X_{n+1} (where X_i is S_i or $\#S_i$ according to whether V_i is $\mathbf{1}$ or $\mathbf{0}$); and due to Theorem 3, $X_1,...,X_n$ entail X_{n+1} iff X_1►(...(X_n►X_{n+1})...) is valid.

The space of acceptable valuations of a language is normally specified in a metalanguage. However, if the language we are considering is normal, it allows us to express the needed constraints also in the object language itself – hence normal languages can be seen as displaying certain 'self-explicitating' capabilities.

Moreover, the fact that having an implication and a standard negation we can turn every constraint into a positive and absolute one may seem to imply that any set of valuations which is delimitable in terms of constraints at all is delimitable in terms of positive and absolute constraints (namely of the internalizations of the original constraints). This would solve our task. But unfortunately it is hopeless.

The reason is that constraints are 'internalizable' (i.e. expressible in the form of valid sentences) *only if there is a negation and an implication*, where the negation and implication are themselves defined in terms of (non-positive and non-absolute) constraints – and these constraints' ability to constitute negation and implication does not survive their internalization. To see this, consider the definition of negation:

$S^{\mathbf{1}} \models \#S^{\mathbf{0}}$, and
$S^{\mathbf{0}} \models \#S^{\mathbf{1}}$.

These constraints are internalized to

S►$\#\#S$, and
$\#S$►$\#S$,

respectively. However, it is clear that the validity of the internalizations alone does not make # into a negation: this can be seen, e.g., from the fact that they are obviously compatible with # being the identity mapping.

This means that a set of constraints delimiting the space of acceptable valuations cannot be always turned into a set of positive constraints. (Of course not: positive constraints can characterize acceptable valuations only in terms of the setences they must verify, and consequently a valuation which verifies more sentences than some acceptable one is bound to be acceptable too. In contrast to this, non-positive constraints can characterize accpetable valuations also in terms of sentences which are to be *falsified*; and hence they can render a valuation unacceptable even if the set of sentences it verifies contains the set of sentences verified by an acceptable valuation).

However, despite of the fact that the possibility of delimiting the space of acceptable valuations of sentences in terms of positive constraints does not obtain in general, and does not obtain even for strongly normal languages, for a language of the latter kind we can define a language with its space of acceptable valuations delimited by means of merely positive constraints and such that its set of valid sentences coincides with that of the original one. This is what we are going to show now. But let us first adopt some definitions which will allow us talk more concisely:

DEFINITION 8. A language is called (*positively*) *delimited* iff the space of the acceptable valuations of its sentences is the set of all and only valuations satisfying a set of (positive) constraints.

Hence what we are after now is to prove that to every strongly normal and delimited language there exists a positively delimited language with the same class of valid sentences. First we will show that for every strongly normal and delimited language L there exists a certain language L^* with the following two properties:

(*) Whenever $S_1^{V1}, ..., S_n^{Vn} \models S_{n+1}^{Vn+1}$ in L, the internalization of this constraint is valid in L^*.
(**) All constraints delimiting the space of acceptable valuations of L^* are in force in L.

It is easy to see that (*) implies that the set of valid sentences of L is contained in that of L^*; whereas (**) implies that, vice versa, the set of valid sentences of L^* is contained in that of L. To be able to prove (*), we first need to precisely characterize the class of constraints mentioned in it:

DEFINITION 9. Let L be a language and $S_1, ..., S_n, S_{n+1}$ its sentences and $V_1, ..., V_n, V_{n+1}$ truth values. If $S_1^{V1}, ..., S_n^{Vn}$ forces S_{n+1}^{Vn+1}, we will say that the constraint

$$S_1^{V1}, ..., S_n^{Vn} \models S_{n+1}^{Vn+1}$$

is *in force for L*.

Note that if the space of accepable valuations of L is delimited by a set C of constraints, then all constraints from C are in force for L, but not necessarily every constraint which is in force for L belongs to C. For example the constraint

$$S \models S$$

is in force for every language, independently of how its space of acceptable valuations is delimited. Or if the set of delimiting constraints contains

$$S_1^{V1}, ..., S_n^{Vn} \models S^V$$

then also the constraint

$$S_1^{V1}, ..., S_n^{Vn}, S_{n+1}^{Vn+1} \models S^V$$

is thereby in force. Etc. (Hence constraints ‘entail’ other constraints.)

Now we are ready to formulate what we will call the *Internalization Theorem* and what spells out that for a strongly normal and delimited language there is a language which fulfills the above requirement (*). Because its proof is somewhat complicated, we will postpone it to the next section.

THEOREM 6. ('Internalization Theorem') Let L be a language, ► and # be a binary resp. a unary function from the set of sentences of L to the same set, and let the space of acceptable valuations of L be delimited by (MI1), (MI2), (MI3), (N1), (N2) and a set C of other constraints (which implies that the language is strongly normal). Let L^* be a language which is just like L with the exception that its space of acceptable valuations is delimited by the set of constraints containing the (►,*#*)-internalizations of all constraints from C plus the following positive constraints:

(A1) $\models S_2 \blacktriangleright (S_1 \blacktriangleright S_2)$

(A2) $\models (\#S_2 \blacktriangleright \#S_1) \blacktriangleright (S_1 \blacktriangleright S_2)$

(A3) $\models (S_1 \blacktriangleright S_2) \blacktriangleright ((S_2 \blacktriangleright S_3) \blacktriangleright (S_1 \blacktriangleright S_3))$

(A4) $\models (S_1 \blacktriangleright (S_1 \blacktriangleright S_2)) \blacktriangleright (S_1 \blacktriangleright S_2)$

(MP) $S_1 \blacktriangleright S_2, S_1 \models S_2$

Then the (►,*#*)-internalization of every constraint which is in force in L is valid in L^*.
PROOF: Postponed to the next section.

It is obvious that (A1)–(A4) together with (MP) constitute an axiomatization of the classical propositional calculus (this very one being proposed, e.g., by Tarski, 1965); and the theorem resembles the completness theorem for this calculus. However, as we have already indicated, whereas the concept of completness aims at mere coincidence of valid sentences, our aim is more ambitious: to inspect the conditions of coincidence of the entire spaces of acceptable valuations. Classical completness then falls out of this as a special case.

Consider the language of the classical propositional calculus based on the primitive connectives ¬ and → (hereafter ***CPC***). The space of acceptable valuations of its sentences of is obviously delimited by the following constraints

$S^{\mathbf{1}} \models \neg S^{\mathbf{0}}$

$S^{\mathbf{0}} \models \neg S^{\mathbf{1}}$;

$S_1^{\mathbf{0}} \models (S_1 \rightarrow S_2)^{\mathbf{1}}$

$S_2^{\mathbf{1}} \models (S_1 \rightarrow S_2)^{\mathbf{1}}$

$S_1^{\mathbf{1}}, S_2^{\mathbf{0}} \models (S_1 \rightarrow S_2)^{\mathbf{0}}$

Hence the function mapping S on $\neg S$ is a standard negation, whereas that mapping S_1 and S_2 on $S_1 \rightarrow S_2$ is a material implication (and hence ***CPC*** is strongly normal).

Now take the language ***CPC**** which is just like ***CPC*** with the single exception that the space of acceptable valuations of its sentences is delimited by the following (positive) constraints:

(A1^{PC}) $\models S_2 \rightarrow (S_1 \rightarrow S_2)$

(A2^{PC}) $\models (\neg S_1 \rightarrow \neg S_2) \rightarrow (S_2 \rightarrow S_1)$

(A3^{PC}) $\models (S_1 \rightarrow S_2) \rightarrow ((S_2 \rightarrow S_3) \rightarrow (S_1 \rightarrow S_3))$

(A4^{PC}) $\models (S_1 \rightarrow (S_1 \rightarrow S_2)) \rightarrow (S_1 \rightarrow S_2)$

(MP^{PC}) $S_1 \rightarrow S_2, S_1 \models S_2$

According to the theorem just stated, the internalizations of all constraints which are in force for ***CPC*** are valid sentences of ***CPC****. Hence, as, e.g.,

$$S_1 \rightarrow S_2, \neg S_1 \rightarrow S_2 \models S_2$$

is a constraint which is in force in ***CPC*** (as is easily computed), its internalization

$$(S_1 \rightarrow S_2) \rightarrow ((\neg S_1 \rightarrow S_2) \rightarrow S_2)$$

is a valid sentence of ***CPC****. In particular, every sentence which is valid in ***CPC*** is valid in ***CPC****.

Note that it would be quite easy to find *some* positively determined language in which all the internalizations of the constraints being in force in L would be valid – take the trivial language which results from L by replacing its delimitation of the space of acceptable valuations by the single constraint $\models S^{\mathbf{1}}$ (i.e. which would render *all* sentences valid). (It would be easy to find a complete axiomatization of a logic, if we were not to require soundness.) The following lemma says that L^* fulfils not only (*), but also (**); namely that all the constraints delimiting L^* are in force in L.

LEMMA 1. Let L be a strongly normal language with I being its material implication and **#** being its standard negation. Then the constraints (A1)-(A4) and (MP) are in force in L.
PROOF: An easy computation.

The desired result, namely that the set of valid sentences of L^* is the same as that of L is now forthcoming:

THEOREM 7. The sets of valid sentences of the languages L and L^* from Theorem 6 coincide.
PROOF: Let S be valid in L. This means that the constraint $\models S$ is in force in L, and hence its internalization is valid in L^*. But the internalization of $\models S$ is S and hence S is valid in L^*. Let now conversely, S be valid in L^*. As all the costraints of L^* are in force in L (as demonstrated by the previous lemma), everything which is valid in L^* is *a fortiori* valid in L – hence S is valid in L.

This implies that the set of valid sentences of ***CPC*** coincides with that of ***CPC****. However, note that this does *not* mean that the classes of acceptable valuations of the two languages coincide! Indeed, there are valuations which are not acceptable in ***CPC***, but are acceptable in ***CPC****: for example the valuation which maps every sentence on **1** is not acceptable in ***CPC*** (because it violates the constraint $S^{\mathbf{1}} \models \neg S^{\mathbf{0}}$), whereas it is acceptable in ***CPC**** (as the constraints of ***CPC*** are positive, it can force no sentence to be false).

Theorem 7 says that if we are interested only in the validity of sentences, we are free to replace L with L^*. Now we prove that if we 'manually' tamper with the space of the acceptable valuations of L^* so that **#** becomes a negation, the equivalence extends to countervalidity.

LEMMA 2. If (A1)-(A4) and (MP) are in force in L, then no acceptable valuation verifies both S and $\#S$, unless it verifies every sentence whatsoever; i.e. the constraint

$$\#S_1, S_1 \models S_2$$

is in force in L.

PROOF: In force of (A1), every sentence of the form $\#S_1\blacktriangleright(\#S_2\blacktriangleright\#S_1)$ is valid. However, the antecedent of a sentence of the shape $(\#S_1\blacktriangleright(\#S_2\blacktriangleright\#S_1))\blacktriangleright(((\#S_2\blacktriangleright\#S_1)\blacktriangleright(S_1\blacktriangleright S_2))\blacktriangleright(\#S_1\blacktriangleright(S_1\blacktriangleright S_2)))$, which is valid in force of (A3), is just of this form; hence, in force of (MP), every sentence of the form $((\#S_2\blacktriangleright\#S_1)\blacktriangleright(S_1\blacktriangleright S_2))\blacktriangleright(\#S_1\blacktriangleright(S_1\blacktriangleright S_2))$ is valid. However, the antecedent of such sentence is valid in force of (A2), hence, in force of (MP), every sentence of the form $\#S_1\blacktriangleright(S_1\blacktriangleright S_2)$ is valid. This means that if $\#S_1$ is true, also $S_1\blacktriangleright S_2$ must be true; hence that if $\#S_1$ and S_1 are true, S_2 must be true.

THEOREM 8. Let L and L^* be as before. Let L^{**} be a language which is just like L^* with the single possible exception that its space of acceptable valuation excludes the valuation which falsifies every sentence. Then both the sets valid and the sets of countervalid sentences of L and L^{**} coincide.
PROOF: It follows from (N1) that the space of acceptable valuations of L does not contain the valuation falsifying all sentences. Hence every acceptable valuation of L is an acceptable valuation of L^{**}; and hence every sentence countervalid in L^{**} is countervalid in L.
From the other side, all the internalizations of the constraints of L clearly keep to be valid in L^{**}, and hence if a sentence S is countervalid in L, $\#S$ is bound to be valid in L^{**}. Moreover, no valuation of L^{**} verifies both S and $\#S$ (due to the previous lemma) – i.e. $\#$ is a negation. This means that if $\#S$ valid, S is bound to be countervalid (*cf.* Theorem 2(i)). It follows that everything which is countervalid in L is countervalid in L^{**}.

Let ***CPC***** be like ***CPC**** with the single exception that its space of acceptable valuations does not contain the function mapping everything on **1**. Then both the spaces of valid and countervalid sentences of ***CPC***** and ***CPC*** coincide. Note that this still does not mean that the classes of acceptable valuations of the two languages coincide. For example, a valuation which falsifies both $\mathbf{S_1}$ and $\neg\mathbf{S_1}$ is not acceptable in ***CPC*** (because it violates the constraint $S^{\mathbf{0}} \models \neg S^{\mathbf{1}}$), but it is acceptable in ***CPC***.

If we now, moreover, guarantee that **#** is a *standard* negation, the equivalence becomes complete:

THEOREM 9. Let L and L^{**} be as before. Let L^{***} be a language which is just like L^{**} with the single possible exception that its space of acceptable valuations excludes all valuations which, for some sentence S, falsity both S and $\#S$. Then the spaces of acceptable valuations of L and L^{***} coincide, and hence L coincides with L^{***}.
PROOF: It follows from (N2) that no valuation for which there exists a sentence S such that both S and $\#S$ are falsified is acceptable in L. Hence none of the valuations which are acceptable in L^{**}, but not in L^{***} are acceptable in L, and hence every acceptable valuation of L is an acceptable valuation of L^{***}.
From the other side, all the internalizations of the constraints of L still keep to be valid in L^{***}. Moreover, every acceptable valuation of L^{***} now verifies S if and only if it falsifies $\#S$ and vice versa – i.e. **#** is a standard negation. This means that the validity of an internalization implies the internalized constraint's being in force (*cf.* Theorem 3(i)); and hence that every constraint which is in force in L is in force also in L^{***}. It follows that every acceptable valuation of L^{***} is an acceptable valuation of L.

Let ***CPC****** be like ***CPC***** with the single exception that its space of acceptable valuations does not contain any function falsifying any sentence together with its negation. Then the spaces of acceptable valuations of ***CPC****** and ***CPC*** coincide, and hence so do the two languages.

Proof of the 'Internalization Theorem'

The proof of the basic theorem of the previous section is somewhat complex, and hence in order not to blur the coherence of the exposition it had to be postponed. We are going to present it in this section. Remember that what we want to prove is that for every strongly normal language whose valuations are delimited by a class *C* of constraints there is a certain language with merely positive constraints such that the internalizations of all constraints from *C* are valid in it. In order to be able to prove this, we must first characterize the class of all constraints which are in force in a language whose space of acceptable valuations is delimited by *C*.

Hence the question which is before us is the following: given that the space of acceptable valuations of a language *L* is delimited by the class *C* of constraints, what is the set of all constraints which are in force for *L* (which are 'entailed' by *C*)? The strategy we are going to employ answering it is based on the fact that what a constraint $S_1^{V1}, ..., S_n^{Vn} \models S^V$ effects is the exclusion of all valuations of the sentences of *L* which map S_i on V_i, but do not map *S* on *V* – in other words which (being subsets of the Cartesian product of the set of sentences of *L* and the set $\boldsymbol{B}$ of the two truth values) contain the set $\{<S_1,V_1,>,...,<S_n,V_n,>,<S,V^C,>\}$ where V^C is the truth-value complementary to *V*. Now we are first going to characterize the set of all valuations not containing any of a given set of subsets of $S\times\boldsymbol{B}$, and then go on to transform this characterization, in several steps, into the desired characterization of all constraints 'entailed' by a given class of constraints.

DEFINITION 10. Let *S* be the set of sentences of a language. A *truth-valuative set* (*over S*) (*tvs*) will be any subset of $S\times\boldsymbol{B}$. If $Y = <S,V>$ is an element of $S\times\boldsymbol{B}$, then Y^C will denote the pair $<S,V^C>$, where V^C is the truth value complementary to *V*. A tvs is called an *elementary contradiction* if it is of the form $\{Y, Y^C\}$. A tvs is called a (*truth-*)*valuation* (*tv*) (*of S*) if it is a total function. A tvs is called inconsistent iff it is contained in no tv.

LEMMA 3. A tvs is inconsistent iff it contains some *Y* together with Y^C; hence the set of all inconsistent tvs's is the smallest set containing all elementary contradictions and closed to forming supersets.
PROOF: Obvious.

DEFINITION 11. If *C* is a set of tvs's, then a tvs is *C-inconsistent* if it is contained in no tv which does not have an element of *C* for its subset.

It might seem that *C*-inconsistent tvs's are precisely those which contain either an elementary contradiction or an element of *C*. But a more careful consideration reveals that this is not the case. Consider a tvs *X* and imagine we want to extend it to a tv not containing an element of *C*. The extension is clearly impossible if *X* contains an elementary contradiction or an element of *C*. Otherwise we can imagine that the extension proceeds in the following way: we take one of the elements of *S* which does not belong to the domain of *X* (i.e. such that $<S,V>$ does not belong to *X* for any *V*), form the pair $<S,V>$, where *V* is an arbitrarily chosen truth value, and add it to *X*; and we repeat this for all the elements of the complement of the domain of *X* in *S*. Could such a procedure fail to provide the desired extension? Indeed it could: it might happen that both $X\cup\{<S,\mathbf{1}>\}$ and $X\cup\{<S,\mathbf{0}>\}$ belong to *C* (or an analogous thing may happen at some later step). In such a case X is not extendable to a tv not containing an element of C; and hence is C-inconsistent. In other words, we have another closure condition for the set of C-inconsistent sets: X is C-inconsistent if both $X\cup\{<S,\mathbf{1}>\}$ and $X\cup\{<S,\mathbf{0}>\}$ are C-inconsistent.

DEFINITION 12. The tvs X is a *resolution* of the tvs's Y_1 and Y_2 iff there is a sentence S and a truth value V so that $Y_1=X\cup\{<S,V>\}$ and $Y_2=X\cup\{<S,V^C>\}$.

LEMMA 4. The set of C-inconsistent tvs's is the smallest set containing all elementary contradictions and all elements of C, and closed to forming supersets and resolutions.
PROOF: We have seen that the only tvs which cannot be extended to a tv is one which either contains an elementary contradiction, or contains an element of C, or is the resolution of two C-inconsistent tvs's.

DEFINITION 13. A *quasiconstraint* (*qc*) is an ordered pair $<X,Y>$ where X is a tvs and Y is an element of $S\times\boldsymbol{B}$. A tv is said to *conform* to the qc if it either does not contain X or contains Y. A qc is said to be *valid* if it is conformed to by every tv. If C is a set of qc's, then a qc is said to be C-valid if every tv which conforms to every element of C conforms to it. The qc $<X,Y^C>$ is said to be a *representation* of the tvs $X\cup Y$; $X\cup Y^C$ is said to be the *projection* of $<X,Y>$. Two qc's are *equivalent* iff they have the same projection.

LEMMA 5. A qc is valid iff its projection is inconsistent. It is C-valid iff its projection is C^*-inconsistent (where C^* is the set of all projections of elements of C).
PROOF: Obvious.

It is clear that a set contains all C-valid qc's iff: it contains representations of all C^*-inconsistent sets and it is closed to the equivalence of qc's (i.e. iff it always contains all constraints equivalent to a constraint it contains).

LEMMA 6. A set contains all C-valid qc's iff:

(i) it contains $<\{Y\},Y>$ for every Y
(ii) it contains all elements of C
(iii) it contains $<X\cup\{Y\},S>$ whenever it contains $<X,S>$
(iv) it contains $<X,S>$ whenever it contains $<X\cup\{Y\},S>$ and $<X\cup\{Y^C\},S>$
(v) it contains $<\{X_1,...,X_{i-1},Y^C,X_{i+1},...,X_n\},X_i^C>$ whenever it contains $<\{X_1,...,X_n\},Y>$

PROOF: (i) guarantees that the set contains representations of all elementary contradictions; (ii) guarantees that it contains representations of all projections of C. (iii) guarantees that it contains a representation of a tvs, then it contains a representation of every its superset; whereas (iv) guarantees that if it contains representations of two tvs's which have a resolution, then it contains a representation of the resolution. (v) then guarantees that if the set contains a representation of a tvs, then in contains all other representations of the same tvs.

DEFINITION 14. A *constraint* (*c*) is an ordered pair $<X,Y>$ where X is a n-tuple of elements of $S\times\boldsymbol{B}$ and Y is an element of $S\times\boldsymbol{B}$. A c $<<X_1,...,X_n>,Y>$ is said to *represent* the qc $<\{X_1,...,X_n\},Y>$; whereas the latter is said to be a *projection* of the former. Two constraints are said to be *equivalent* if their projections coincide.

LEMMA 7. A set contains all C-valid c's iff:

(i) it contains $<<Y>,Y>$ for every Y

(ii) it contains all elements of C

(iii) it contains $<<X_0,X_1,...,X_n>,Y>$ whenever it contains $<<X_1,...,X_n>,Y>$

(iv) it contains $<<X_2,...,X_n>,Y>$ whenever it contains $<<X_1,...,X_n>,Y>$ and $<<X_1^C,...,X_n>,Y>$

(v) it contains $<<Y^C,X_2,...X_n>,X_1^C>$ whenever it contains $<<X_1,...,X_n>,Y>$

(vi) it contains $<<X_{\pi(1)},...,X_{\pi(n)}>,Y>$ whenever it contains $<<X_1,...,X_n>,Y>$, for every permutation π of $\{1,...,n\}$

(vii) it contains $<<X_1,...,X_n>,Y>$ whenever it contains $<<X_1,X_1,...,X_n>,Y>$

PROOF: A c is C-valid iff it is a representation of a C^*-valid qc, where C^* is the set of projection of the elements of C. Hence a set contains all C-valid c's iff it contains a representation of every C^*-valid qc and it contains every c equivalent to a c it contains. The former condition is guaranteed by (i)-(v) which straightforwardly correspond to the clauses of the previous lemma. The latter is guaranteed by (vi) and (vii) (that the set contains $<<X_1,X_1,...,X_n>,Y>$ whenever it contains $<<X_1,...,X_n>,Y>$ follows from (iii)).

Now we can pass over from constraints to sentences (remember that the (►,#-)*internalization* of the constraint $S_1^{V1}, ..., S_n^{Vn} \models S_{n+1}^{Vn+1}$ is the sentence $(X_1►...(X_n►X_{n+1}))$, where X_i is S_i if V_i is **1** and is $\#S_i$ if V_i is **0**)

LEMMA 8. A set of sentences contains the ►,#-internalizations of all C-valid c's iff:

(i) it contains $(S►S)$ for every sentence S

(ii) if $c \in C$, then it contains the ►,#-internalization of c

(iii) it contains $X_0►(X_1►...(X_n►Y)...)$ whenever it contains $X_1►(...(X_n►Y)...)$

(iv) it contains $X_2►(...(X_n►Y)...)$ whenever it contains $X_1►(X_2►...(X_n►Y)...)$ and $\#X_1►(X_2►...(X_n►Y)...)$

(v) it contains $X_1►(...(\#Y►\#X_n)...)$ whenever it contains $X_1►(...(X_n►Y)...)$

(vi) it contains $(X_{\pi(1)}►(...(X_{\pi(n)}►Y)...)$ whenever it contains $X_1►(...(X_n►Y)...)$, for every permutation π of $\{1,...,n\}$

(vii) it contains $X_1►(...(X_n►Y)...)$ whenever it contains $X_1►(X_1►...(X_n►Y)...)$ and *vice versa*.

PROOF: Each of the clauses of the lemma obviously guarantees that the set contains the internalizations of all c's specified in the corresponding clause of the previous lemma.

LEMMA 9. A set of sentences contains the internalizations of all C-valid c's iff it contains all sentences of the following shapes:

(i) $X►X$

(ii) the ►,#-internalization of c, where $c \in C$

(iii) $(X_1►...(X_n►Y))►(X_0►(X_1►...(X_n►Y)...))$

(iv) $(X_1►(X_2►...(X_n►Y)))►((\#X_1►(X_2►...(X_n►Y)))►(X_2►...(X_n►Y)))$

(v) $((X_1►...(X_n►Y)))►(X_1►...(\#Y►\#X_n))))$

(vi) $(X_1►...(X_n►Y))►(X_{\pi(1)}►...(X_{\pi(1)}►Y))$ for every permutation π of $\{1,...,n\}$

(vii) $((X_1►(X_1►...(X_n►Y))))►(X_1►...(X_n►Y))))$

and

(viii) it contains Y whenever it contains $(X►Y)$ and X

PROOF: Given (viii), (iii)-(vii) obviously have the power of the corresponding clauses of the previous lemma.

LEMMA 10. Let a set of sentences contain all sentences of the following shapes:

(ii) $Y▶(X▶Y)$

(iii) $(\#Y▶\#X)▶(X▶Y)$

(iv) $(X_1▶X_2)▶((X_2▶X_3)▶(X_1▶X_3))$

(v) $(X▶(X▶Y))▶(X▶Y)$

and let it

(vi) contain Y whenever it contains $X▶Y$ and X.

Then the set contains all sentences of the following shapes:

(a) $Y▶Y$

(b) $X_1▶((X_1▶X_2)▶X_2)$

(c) $(X_1▶(X_2▶X_3))▶(X_2▶(X_1▶X_3))$

(d) $\#X_1▶(X_1▶X_2)$

(e) $\#\#X_1▶X_1$

(f) $X_1▶\#\#X_1$

(g) $(X_2▶X_3)▶((X_1▶X_2)▶(X_1▶X_3))$

(h) $(X▶Y)▶(\#Y▶\#X)$

(i) $(X_1▶X_2)▶((\#X_1▶X_2)▶X_2)$

(j) $X_1▶(\#X_2▶\#(X_1▶X_2))$

PROOF: As the statements (ii)-(v) and the rule (vi) straightforwardly correspond to an axiomatization of the classical propositional calculus (the Tarski's one mentioned above), the proof is straightforwardly analogous to the proof of the fact that statements corresponding to (a) – (j) are provable within the system.
In order to make it more comprehensible, let us adopt the following notational conventions: A formula following by the symbol n in brackets is a shorthand for 'F belongs to the set for it is of the form n' (where n is either a letter marking a formula proved earlier or a number formula occuring earlier in the proof). A formula followed by two symbols n_1 and n_2 in brackets is a shorthand for 'F belongs to the set in force of (vi) applied to the formulas n_1 and n_2'. Moreover, we will make use of the fact that if the set contains a formula $(X_1▶...(X_n▶Y))$, then if it contains its 'antecedents' X_1, ..., X_n, it is, in force of (iv) bound to contain Y: hence a formula followed by a letter L and symbols n_1 through n_n in brackets will be a shorthand for 'F belongs to the set in force of the set containing L and its 'antecedents' n_1 through n_2'.
Now we can give the proofs of the clauses (a)-(j) in the following concise form:

a:

1 (ii) $(Y▶(Y▶Y))$

2 (v 1) $(Y▶Y)$

b:

1 (ii) $X_1▶((X_1▶X_2)▶X_1)$

2 (iv) $((X_1▶X_2)▶X_1)▶(X_1▶X_2)▶((X_1▶X_2)▶X_2)$

3 (iv 1,2) $X_1▶(X_1▶X_2)▶((X_1▶X_2)▶X_2)$

4 (v) $(X_1▶X_2)▶((X_1▶X_2)▶X_2)▶((X_1▶X_2)▶X_2)$

5 (iv 3,4) $X_1▶((X_1▶X_2)▶X_2)$

c:

1 (iv) $(X_1 ▶ (X_2 ▶ X_3)) ▶ (((X_2 ▶ X_3) ▶ X_3) ▶ (X_1 ▶ X_3))$
2 (iv) $X_2 ▶ ((X_2 ▶ X_3) ▶ X_3) ▶ ((((X_2 ▶ X_3) ▶ X_3) ▶ (X_1 ▶ X_3)) ▶ (X_2 ▶ (X_1 ▶ X_3)))$
3 (a) $X_2 ▶ ((X_2 ▶ X_3) ▶ X_3)$
4 (2,3) $((((X_2 ▶ X_3) ▶ X_3) ▶ (X_1 ▶ X_3)) ▶ (X_2 ▶ (X_1 ▶ X_3)))$
5 (iv 1,4) $((X_1 ▶ (X_2 ▶ X_3)) ▶ (X_2 ▶ (X_1 ▶ X_3)))$
d:
1 (ii) $(\#X_1 ▶ (\#X_2 ▶ \#X_1))$
2 (iii) $((\#X_2 ▶ \#X_1) ▶ (X_1 ▶ X_2))$
3 (iv 1,2) $(\#X_1 ▶ (X_1 ▶ X_2))$
e:
1 (c) $(\#\#X_1 ▶ (\#X_1 ▶ \#X_2))$
2 (iii) $((\#X_1 ▶ \#X_2) ▶ (X_2 ▶ X_1))$
3 (iv 1,2) $(\#\#X_1 ▶ (X_2 ▶ X_1))$
4 (c 3) $(X_2 ▶ (\#\#X_1 ▶ X_1))$
5 (4) $((X_2 ▶ (\#\#X_1 ▶ X_1)) ▶ (\#\#X_1 ▶ X_1))$
6 (4,5) $(\#\#X_1 ▶ X_1)$
f:
1 (d) $(\#\#\#X_1 ▶ \#X_1)$
2 (iii) $((\#X_1 ▶ \#X_1) ▶ (X_1 ▶ X_1))$
3 (1,2) $(X_1 ▶ \#\#X_1)$
g:
1 (iv) $(X_1 ▶ X_2) ▶ ((X_2 ▶ X_3) ▶ (X_1 ▶ X_3))$
2 (b) $(X_1 ▶ X_2) ▶ ((X_2 ▶ X_3) ▶ (X_1 ▶ X_3)) ▶ (X_2 ▶ X_3) ▶ ((X_1 ▶ X_2) ▶ (X_1 ▶ X_3))$
3 (1,2) $(X_2 ▶ X_3) ▶ ((X_1 ▶ X_2) ▶ (X_1 ▶ X_3))$
h:
1 (iv) $(\#\#X_1 ▶ \#\#X_2) ▶ ((\#\#X_2 ▶ X_2) ▶ (\#\#X_1 ▶ X_2))$
2 (c 1) $(\#\#X_2 ▶ X_2) ▶ ((\#\#X_1 ▶ \#\#X_2) ▶ (\#\#X_1 ▶ X_2))$
3 (e) $(\#\#X_2 ▶ X_2)$
4 (2,3) $((\#\#X_1 ▶ \#\#X_2) ▶ (\#\#X_1 ▶ X_2))$
5 (iv) $(X_1 ▶ \#\#X_1) ▶ ((\#\#X_1 ▶ X_2) ▶ (X_1 ▶ X_2))$
6 (f) $(X_1 ▶ \#\#X_1)$
7 (5,6) $((\#\#X_1 ▶ X_2) ▶ (X_1 ▶ X_2))$
8 (iv 4,7) $((\#\#X_1 ▶ \#\#X_2) ▶ (X_1 ▶ X_2))$
i:
1 (iv) $(\#X_2 ▶ \#X_1) ▶ ((\#X_1 ▶ (X_1 ▶ X_3)) ▶ (\#X_2 ▶ (X_1 ▶ X_3)))$
2 (c 1) $(\#X_1 ▶ (X_1 ▶ X_3)) ▶ ((\#X_2 ▶ \#X_1) ▶ (\#X_2 ▶ (X_1 ▶ X_3)))$
3 (d) $(\#X_1 ▶ (X_1 ▶ X_3))$
4 (3,2) $((\#X_2 ▶ \#X_1) ▶ (\#X_2 ▶ (X_1 ▶ X_3)))$
5 (iv) $(\#X_2 ▶ X_1) ▶ ((X_1 ▶ (\#X_2 ▶ X_3)) ▶ (\#X_2 ▶ (\#X_2 ▶ X_3)))$
6 (c 5) $(X_1 ▶ (\#X_2 ▶ X_3)) ▶ ((\#X_2 ▶ X_1) ▶ (\#X_2 ▶ (\#X_2 ▶ X_3)))$
7 (c) $((\#X_2 ▶ (X_1 ▶ X_3)) ▶ (X_1 ▶ (\#X_2 ▶ X_3)))$
9 (iv 6,7) $(\#X_2 ▶ (X_1 ▶ X_3)) ▶ ((\#X_2 ▶ X_1) ▶ (\#X_2 ▶ (\#X_2 ▶ X_3)))$
10 (v) $((\#X_2 ▶ (\#X_2 ▶ X_3)) ▶ (\#X_2 ▶ X_3)))$
11 (iv) $((\#X_2 ▶ X_1) ▶ (\#X_2 ▶ (\#X_2 ▶ X_3))) ▶ (\#X_2 ▶ (\#X_2 ▶ X_3)) ▶ (\#X_2 ▶ X_3)),((\#X_2 ▶ X_1) ▶ (\#X_2 ▶ X_3))))$
12 (iv 9,11) $(\#X_2 ▶ (X_1 ▶ X_3)) ▶ ((\#X_2 ▶ X_1) ▶ (\#X_2 ▶ X_3))$

13 (iv 4,12) $(\#X_2 \blacktriangleright \#X_1) \blacktriangleright ((\#X_2 \blacktriangleright X_1) \blacktriangleright (\#X_2 \blacktriangleright X_3))$
14 (h) $((X_1 \blacktriangleright X_2) \blacktriangleright (\#X_2 \blacktriangleright \#X_1))$
15 (iv 14,15) $(X_1 \blacktriangleright X_2) \blacktriangleright ((\#X_1 \blacktriangleright X_2) \blacktriangleright (\#X_3 \blacktriangleright X_2))$
16 (15) $(X_1 \blacktriangleright X_2) \blacktriangleright ((\#X_1 \blacktriangleright X_2) \blacktriangleright (\#\#(X_1 \blacktriangleright X_1) \blacktriangleright X_2))$
17 (a) $(\#\#(X_1 \blacktriangleright X_1) \blacktriangleright X_2) \blacktriangleright (\#\#(X_1 \blacktriangleright X_1) \blacktriangleright X_2)$
18 (c 17) $\#\#(X_1 \blacktriangleright X_1) \blacktriangleright ((\#\#(X_1 \blacktriangleright X_1) \blacktriangleright X_2) \blacktriangleright X_2)$
19 (a) $(X_1 \blacktriangleright X_1)$
20 (f 19) $\#\#(X_1 \blacktriangleright X_1)$
21 (20,18) $\#\#(X_1 \blacktriangleright X_1) \blacktriangleright ((\#\#(X_1 \blacktriangleright X_1) \blacktriangleright X_2) \blacktriangleright X_2)$
22 (18,21) $((\#\#(X_1 \blacktriangleright X_1) \blacktriangleright X_2) \blacktriangleright X_2)$
23 (g 22) $((\#X_1 \blacktriangleright X_2) \blacktriangleright (\#\#(X_1 \blacktriangleright X_1) \blacktriangleright X_2)) \blacktriangleright ((\#X_1 \blacktriangleright X_2) \blacktriangleright X_2)$
24 (iv 15,23) $(X_1 \blacktriangleright X_2) \blacktriangleright ((\#X_1 \blacktriangleright X_2) \blacktriangleright X_2)$
j:
1 (a) $((X_1 \blacktriangleright X_2) \blacktriangleright (X_1 \blacktriangleright X_2))$
2 (c) $((X_1 \blacktriangleright X_2) \blacktriangleright (X_1 \blacktriangleright X_2)) \blacktriangleright (X_1 \blacktriangleright ((X_1 \blacktriangleright X_2) \blacktriangleright X_2))$
3 (1,2) $X_1 \blacktriangleright ((X_1 \blacktriangleright X_2) \blacktriangleright X_2)$
4 (h) $(((X_1 \blacktriangleright X_2) \blacktriangleright X_2) \blacktriangleright (\#X_2 \blacktriangleright \#(X_1 \blacktriangleright X_2)))$
5 (iv 3,4) $(X_1 \blacktriangleright (\#X_2 \blacktriangleright \#(X_1 \blacktriangleright X_2)))$

LEMMA 11. A set of sentences contains the representations of all C-valid c's iff it contains all sentences of the following shapes::

(i) the f,g-internalization of c, where $c \in C$
(ii) $Y \blacktriangleright (X \blacktriangleright Y)$
(iii) $(\#Y \blacktriangleright \#X) \blacktriangleright (X \blacktriangleright Y)$
(iv) $(X_1 \blacktriangleright X_2) \blacktriangleright ((X_2 \blacktriangleright X_3) \blacktriangleright (X_1 \blacktriangleright X_3))$
(v) $(X \blacktriangleright (X \blacktriangleright Y)) \blacktriangleright (X \blacktriangleright Y)$

and

(vi) it contains Y whenever it contains $(X \blacktriangleright Y)$ and X.

PROOF: Let us prove that given (i)-(vi), the individual clauses of the previous lemma are fulfilled. I will distinguish the numbers of the clauses of the previous lemma from those of the current one by putting them into square brackets. The clauses [ii], [iii], [vii] and [viii] are unproblematic, they follow directly from (i), (ii), (v) and (vi). Hence we are left with proving [i], [iv], [v] and [vi]. [i] is Lemma 10(a). [iv] is an instance of Lemma 10(h). As for [v], it follows from (iii) that the set contains $(X_n \blacktriangleright Y) \blacktriangleright (\#Y \blacktriangleright \#X_n)$. Now as it follows from Lemma 10(g) that it contains $(X_1 \blacktriangleright X_2) \blacktriangleright (X_1 \blacktriangleright X_3)$ whenever it contains $(X_2 \blacktriangleright X_3)$, it must contain $(X_{n-1} \blacktriangleright (X_\mathrm{n} \blacktriangleright Y)) \blacktriangleright (X_{n-1} \blacktriangleright (\#Y \blacktriangleright \#X_\mathrm{n}))$, and, by repetitive application of the same step, $((X_1 \blacktriangleright ... (X_\mathrm{n} \blacktriangleright Y))) \blacktriangleright (X_1 \blacktriangleright ... (\#Y \blacktriangleright \#X_\mathrm{n}))$. Now consider [vi]: as the set contains $((X_i \blacktriangleright X_{i+1} \blacktriangleright ... (X_\mathrm{n} \blacktriangleright Y))) \blacktriangleright (X_{i+1} \blacktriangleright (X_i \blacktriangleright ... (X_\mathrm{n} \blacktriangleright Y)))$ which is an instance of the formula in Lemma 10(c), and as it contains $(Y \blacktriangleright X_1) \blacktriangleright (Y \blacktriangleright X_2)$ whenever it contains $(X_1 \blacktriangleright X_2)$ (which follows from Lemma 10(g)), it must contains $((X_{i-1} \blacktriangleright (X_i \blacktriangleright X_{i+1} \blacktriangleright ... (X_\mathrm{n} \blacktriangleright Y)))) \blacktriangleright (X_{i-1} \blacktriangleright (X_{i+1} \blacktriangleright (X_i \blacktriangleright ... (X_\mathrm{n} \blacktriangleright Y))))$; and, by the reperitive application of the same step, it must contain also $(X_1 \blacktriangleright ... (X_i \blacktriangleright (X_{i+1} \blacktriangleright ... (X_\mathrm{n} \blacktriangleright Y))))) \blacktriangleright (X_1 \blacktriangleright ... (X_{i+1} \blacktriangleright (X_i \blacktriangleright ... (X_\mathrm{n} \blacktriangleright Y))))$. And as any permutation of a finite sequence is the result of a finite number of exchanges of consequent elements of the sequence, it contains $(X_1 \blacktriangleright ... (X_n \blacktriangleright Y)) \blacktriangleright (X_{\pi(1)} \blacktriangleright (... (X_{\pi(n)} \blacktriangleright Y)...))$ for every permutation π of $\{1,...,n\}$.

LEMMA 10. Let (A1)-(A4) and (MP) be in force for a language L. Then the ►,#-internalizations of (N1), (N2), (MI1), (MI2), and (MI3) are valid in L.

PROOF: The internalizations are:

(N1^I) S►##S

(N2^I) #S►#S

(MI1^I) #S_1►(S_1►S_2)

(MI2^I) S_2►(S_1►S_2)

(MI3^I) S_1►(#S_2►#(S_1►S_2))

(N1^I) is an instance of Lemma 10(f); (N1^I) is is an instance of Lemma 10(a); (MI1^I) is an instance of Lemma 10(d); (MI2^I) is an instance of (ii); and (MI3^I) is an instance of Lemma 10(i).

Now we are ready to present the proof of the 'Internalization Theorem':

PROOF OF THEOREM 6. Let L be a language, ► and # be a binary resp. a unary function from the set of sentences of L to the same set, and let the space of acceptable valuations of L be delimited by (MI1), (MI2), (MI3), (N1), (N2) and a set C of other constraints. Let L^* be a language which is just like L with the exception that its space of acceptable valuation is delimited by the set of constraints containing the ►,#-internalizations of all constraints from C plus the constraints (A1)-(A4) and (MP). As accodring to the previous lemma the ►,#-internalizations of (MI1), (MI2), (MI3), (N1), (N2) are valid in L^*, the internalizations of all constraints of delimiting the space of acceptable valuations of the sentences of L are valid in L^*. Let C^* be $C\cup\{$(MI1), (MI2), (MI3), (N1), (N2)$\}$; then to be in force in L is to be C^*-valid; and it follows from Lemma 11 that the internalizations of all C^*-valid constraints are valid in L^*.

Conslusion

The upshots of the above considerations is that the relstionship between the semantics and the axiomatics of the predicate calculus is more intricate then it might *prima facie* seem; and that we are not able to fix the denotations of the standard logical constants in terms of axioms. (This is of course, one of the sources of the relative popularity of the intuitionistic logic, whose relationship to axioms and rules is much more straightforward – see Peregrin, 2004). What we have gained as a by-product is a story about why the axioms of an axiomatic system of the classical propositional logic are such as they are: we can read them as outcomes of a proces of articulation of restrictions of acceptability of truth valuations.

References

Carnap, R. (1943): *Formalization of Logic*, Harvard University Press, Cambridge (Mass.).

Kalmár, L. (1936): 'Zurückführung des Entscheidungsproblems auf den Fall von Formeln mit einer einzigen binären Funktionsvariablen', *Comp. math.* 4, 137-144.

Koslow, A. (1992): *A Structuralist Theory of Logic*, Cambridge University Press, Cambridge.

Mendelsohn, E. (1964): *Introduction to Mathematical Logic*, Wadsworth & Brooks, Pacific Grove.

Peregrin, J. (1997): 'Language and its Models', *Nordic Journal of Philosophical Logic* 2, 1-23.

Peregrin, J. (2004): 'Logic as "making it explicit"', *The Logica Yearbook '03*, Filosofia, Praha.
Tarski, A. (1965): *Introduction to Logic and to the Methodology of Deductive Sciences*, Oxford University Press, Oxford.