The Minimal Relevant Logic and the Call-by-Value Lambda Calculus*

S. van Bakel, M. Dezani-Ciancaglini, U. de'Liguoro, Y. Motohama

Department of Computing, Imperial College London, UK
Dipartimento di Informatica, Università degli Studi di Torino, Italy

Abstract

The minimal relevant logic B+, seen as a type discipline, includes an extension of Curry types known as the intersection type discipline. We will show that the full logic B+ gives a type assignment system which produces a model of Plotkin's call-by-value λ-calculus.

1 Introduction

The logical system B+ arose from Meyer and Routley's investigation on the negation-free entailment logic [12]. In their approach, B+ turns out as the minimal relevant logic, which is complete with respect to a variant of Kripke models, called the positive model structures.

Independently, and with quite different aims, an extension of the Curry type assignment system with "intersection" types was introduced in [3] by Coppo and Dezani. The same authors, together with Barendregt, were able to prove in [2] that, provided a suitable axiomatisation of the subtype relation $A \leq B$, the set of filters over types is a λ-model. As a consequence, a completeness theorem for the intersection types assignment system was established.

As remarked by Meyer in [8], it can be recognised that $A \leq B$ holds in the type theory of [2] if and only if, "on translation", $A \to B$ is a theorem of B+. However, the language of B+ has one more connective than intersection types, namely disjunction. Extensions of the intersection type discipline with a "union" type constructor have been pursued, by some of the present authors together with others, both in the framework of classical λ-calculus and in that of some "parallel" and non-deterministic extensions of λ-calculus: see [1, 5, 4]. From this investigations it turns out that strong systems of union types do not give filter λ-models, while weaker systems allow for a satisfactory logical analysis of some extended λ-calculi.

Here we will follow a different path. Instead of extending the calculus, we will consider Plotkin's *call-by-value λ-calculus* [9], whose syntax is the same as that of the λK-calculus, but the β-rule is replaced by a restricted form:

$$(\beta_v) \quad (\lambda x.M)N \longrightarrow M[N/x], \quad \text{if } N \text{ is not an application.}$$

The idea is that a term that is an application needs to be further evaluated before it can be passed on as an argument. By rule (β_v), in contrast to the classical β-rule, substitution is delayed until the evaluation of the argument N reaches a value, namely either a variable or an abstraction. Because of this, the call-by-value λ-calculus has been proposed and studied as the abstract counterpart of call-by-value lazy programming languages, reflecting closer

* Partially supported by NATO Collaborative Research Grant CRG 970285 'Extended Rewriting and Types'.

the actual practice of language implementation. In fact, the subtleties of parameter passing can be caught by the formal calculus, and analyzed using both proof theoretical and model theoretical tools in an elegant way [6, 10, 11].

What we will show here is that B+is the type theory of the call-by-value λ-calculus, in the following sense: we will consider the full language of the logic B+, including disjunction, and take the B+ axioms and rules as defining a subtype relation. Then we will consider a type assignment system which is essentially that of [2], but using the extended type syntax and subtyping relation hinted above, with a key restriction on the types that can be assumed for variables in a basis. It turns out that, within this system, types are preserved under β_v-conversion. Moreover, the filter structure of the type theory is an instance of call-by-value syntactical λ-model (a generalisation, [6, 10, 11] of the notion of syntactical λ-model due to Hindley and Longo [7]).

2 The minimal relevant logic B+ as a type discipline

The minimal relevant logic B+ is a propositional calculus. To see B+as a type discipline, we interpret propositions as types: the constant for truth is interpreted as a constant type ν, whose intended meaning is "the type of values"; implication as the arrow type connective $\rightarrow$, conjunction and disjunction as intersection ($\wedge$) and union type connectives ($\vee$), respectively. Then we define a preorder relation $\leq$ such that $A \leq B$ if and only if $A \rightarrow B$ is a theorem of B+.

Definition 2.1 (The set L of formulae) The set L of formulae is defined by

i) a denumerable set $\mathcal{PV}$ of propositional variables, ranged over by a, b, c possibly with subscripts;

ii) $\nu \in \mathsf{L}$ (a type constant);

iii) $A, B \in \mathsf{L} \Rightarrow (A \rightarrow B), (A \wedge B), (A \vee B) \in \mathsf{L}$.

Notation. To avoid using too many parentheses, we assume that $\wedge$ and $\vee$ take precedence over $\rightarrow$, and that $\rightarrow$ associates to the right.

Definition 2.2 (The minimal relevant logic B+ with $\leq$) B+ is the logic on the language L defined by the following axioms and rules:

$(A1)$: $A \leq \nu$
$(A2)$: $A \leq A$
$(A3)$: $A \leq A \wedge A$
$(A4)$: $A \wedge B \leq A, A \wedge B \leq B$
$(A5)$: $A \vee A \leq A$
$(A6)$: $A \leq A \vee B, B \leq A \vee B$
$(A7)$: $(A \rightarrow B) \wedge (A \rightarrow C) \leq A \rightarrow B \wedge C$
$(A8)$: $(A \rightarrow C) \wedge (B \rightarrow C) \leq A \vee B \rightarrow C$
$(A9)$: $A \wedge (B \vee C) \leq (A \wedge B) \vee (A \wedge C)$
$(R1)$: $A \leq B, B \leq C \Rightarrow A \leq C$ (transitivity)
$(R2)$: $A \leq B, C \leq D \Rightarrow A \wedge C \leq B \wedge D$ ($\wedge$-monotonicity)
$(R3)$: $A \leq B, C \leq D \Rightarrow A \vee C \leq B \vee D$ ($\vee$-monotonicity)
$(R4)$: $A \leq B \Rightarrow B \rightarrow C \leq A \rightarrow C$ (suffixing)
$(R5)$: $B \leq C \Rightarrow A \rightarrow B \leq A \rightarrow C$ (prefixing).

We define $\sim$ as the symmetric closure of $\leq$. The relation $\sim$ enjoys the following properties:

i) $\sim$ is a congruence relation.
ii) $A \wedge B \sim B \wedge A$, $A \vee B \sim B \vee A$.
iii) $(A \wedge B) \wedge C \sim A \wedge (B \wedge C)$, $(A \vee B) \vee C \sim A \vee (B \vee C)$.
iv) $A \wedge (B \vee C) \sim (A \wedge B) \vee (A \wedge C)$, $A \vee (B \wedge C) \sim (A \vee B) \wedge (A \vee C)$ (distributivity).
v) $(A \to B) \wedge (A \to C) \sim (A \to B \wedge C)$.
vi) $(A \to C) \wedge (B \to C) \sim (A \vee B \to C)$.

To further investigate properties of the system B+, it is useful to associate to each type both its conjunctive and disjunctive normal form. First let us define inductively the following subsets of L.

Definition 2.3 (Stratification of L) $T_\to, T_\vee, T_\wedge, T_{\wedge\vee}, T_{\vee\wedge} \subseteq \mathsf{L}$ are inductively defined:
$(T_\to)$: $\nu \in T_\to$; $a \in T_\to$, for all propositional variables a
$A \in T_\wedge, B \in T_\vee \Rightarrow A \to B \in T_\to$
$(T_\vee)$: $A \in T_\to \Rightarrow A \in T_\vee$
$A, B \in T_\vee \Rightarrow A \vee B \in T_\vee$
$(T_\wedge)$: $A \in T_\to \Rightarrow A \in T_\wedge$
$A, B \in T_\wedge \Rightarrow A \wedge B \in T_\wedge$
$(T_{\wedge\vee})$: $A \in T_\vee \Rightarrow A \in T_{\wedge\vee}$
$A, B \in T_{\wedge\vee} \Rightarrow A \wedge B \in T_{\wedge\vee}$
$(T_{\vee\wedge})$: $A \in T_\wedge \Rightarrow A \in T_{\vee\wedge}$
$A, B \in T_{\vee\wedge} \Rightarrow A \vee B \in T_{\vee\wedge}$.

We will now introduce maps from arbitrary types belonging to L into their conjunctive/disjunctive normal forms in $T_{\wedge\vee}$ and $T_{\vee\wedge}$, respectively.

Definition 2.4 The maps $\mathbf{m}_{\wedge\vee} : \mathsf{L} \to T_{\wedge\vee}$ and $\mathbf{m}_{\vee\wedge} : \mathsf{L} \to T_{\vee\wedge}$ are as follows defined by simultaneously induction on the structure of formulae:
i) $\mathbf{m}_{\wedge\vee}(A) = \mathbf{m}_{\vee\wedge}(A) = A$ if $A \in \mathcal{PV} \cup \{\nu\}$.
ii) If $\mathbf{m}_{\vee\wedge}(A) = \bigvee_{i\in I} A_i$ (where each $A_i \in T_\wedge$) and $\mathbf{m}_{\wedge\vee}(B) = \wedge_{j\in J} B_j$ (where each $B_j \in T_\vee$) then
$$\mathbf{m}_{\wedge\vee}(A \to B) = \mathbf{m}_{\vee\wedge}(A \to B) = \wedge_{i\in I} \wedge_{j\in J} (A_i \to B_j).$$
iii) $\mathbf{m}_{\wedge\vee}(A \wedge B) = \mathbf{m}_{\wedge\vee}(A) \wedge \mathbf{m}_{\wedge\vee}(B)$, and, if $\mathbf{m}_{\vee\wedge}(A) = \bigvee_{i\in I} A_i$ and $\mathbf{m}_{\vee\wedge}(B) = \bigvee_{j\in J} B_j$ then
$$\mathbf{m}_{\vee\wedge}(A \wedge B) = \bigvee_{i\in I} \bigvee_{j\in J} (A_i \wedge B_j).$$
iv) $\mathbf{m}_{\vee\wedge}(A \vee B) = \mathbf{m}_{\vee\wedge}(A) \vee \mathbf{m}_{\vee\wedge}(B)$, and, if $\mathbf{m}_{\wedge\vee}(A) = \wedge_{i\in I} A_i$ and $\mathbf{m}_{\wedge\vee}(B) = \wedge_{j\in J} B_j$ then
$$\mathbf{m}_{\wedge\vee}(A \vee B) = \wedge_{i\in I} \wedge_{j\in J} (A_i \vee B_j).$$

We shall prove, in the Appendix, that $A \sim \mathbf{m}_{\wedge\vee}(A) \sim \mathbf{m}_{\vee\wedge}(A)$ for all A.

The Lindenbaum algebra of B+, i.e. the quotient of L under $\sim$, is ordered by the relation $[A]_\sim \leq_{/\sim} [B]_\sim \Leftrightarrow_{\text{def}} A \leq B$ (where $[A]_\sim$ is the equivalence class of A). As an ordered set it is a distributive lattice, where $[A \wedge B]_\sim$ and $[A \vee B]_\sim$ are respectively the meet and the join of $[A]_\sim$ and $[B]_\sim$, and $[\nu]_\sim$ is the top.

Co-prime elements in a lattice, here called $\vee$-prime elements, play a role in the representation theorems of distributive lattices. In our setting, $\vee$-prime formulae will turn out to be types of closed terms, β_v-convertible with values. We introduce them here and defer the analysis of their connection to values to the next sections.

Definition 2.5 A formula A is $\vee$-prime iff $A \leq B \vee C \Rightarrow A \leq B$ or $A \leq C$.

Theorem 2.6 (Properties of $\vee$-prime formulas)

i) *ν and any propositional variable are $\vee$-prime.*

ii) *If $\mathbf{m}_{\vee\wedge}(A) = \bigvee_{i\in I} A_i$ then A_i is $\vee$-prime, for all $i \in I$.*

iii) *The formula $\wedge_{i\in I}(A_i \to B_i)$ is $\vee$-prime for all (finite) I and formulas A_i, B_i.*

iv) *$B \to C \not\sim \nu$ for all B, C.*

v) *$\wedge_{i\in I}(A_i \to B_i) \leq C \to D$ and C is $\vee$-prime imply $C \leq \wedge_{i\in J} A_i$ and $\wedge_{i\in J} B_i \leq D$ for some $J \subseteq I$.*

The proof of Theorem 2.6 is reported in the Appendix. The condition 'C is $\vee$-prime' in statement (v) of the above theorem, is necessary. A counter-example is axiom (A8).

3 The type assignment system

Type assignment systems are formal systems deriving statements of the form $M : A$, where M (the subject) is a pure λ-term and A (the predicate) is a type (here a formula). The intuitive meaning is that types are properties of terms, hence $M : A$ means "M has property A". As M may contain free variables, assumptions about the types of such variables may be used in the derivation, and are collected into *bases* (also called *contexts*): $x_1 : A_1, \ldots, x_n : A_n$, which we consider as finite sets of statements (assumptions) such that $x_i \not\equiv x_j$ if $i \neq j$.

Bases are ranged over by Γ, and we write $\Gamma, x : A$ as abbreviation of $\Gamma \cup \{x : A\}$, under the assumption that x does not occur in Γ. Then the general form of statements derivable in a type assignment system is $\Gamma \vdash M : A$, meaning "M has type A if the variables in Γ have the types listed in Γ itself".

We observe that, although the basic motivation for having bases is to handle the types of the free variables in the subject M, it is not the case that, if $\Gamma \vdash M : A$ is derivable, then the subjects of statements in Γ are exactly the free variables of M. Indeed, on the one hand Γ may contain assumptions about variables not occurring free in M; on the other hand, because of rule (ν), M may contain free variables which are not in Γ.

Definition 3.1 (The type assignment system)

i) A statement is an expression of the form $M : A$, where M is a term (subject) and A is a type (predicate).

ii) An assumption is a statement whose subject is a term variable.

iii) A basis is a set of assumptions with distinct variables as subjects whose predicates are $\vee$-prime types.

iv) A statement $M : A$ is derivable from a basis Γ, notation $\Gamma \vdash M : A$, if $\Gamma \vdash M : A$ can be proved using the following axioms and inference rules:

$$(Ax): \ \frac{}{\Gamma, x : A \vdash x : A}$$

$$(\nu): \ \frac{}{\Gamma \vdash M : \nu} \text{ (if } M \text{ is a variable or an abstraction)}$$

$$(\to I): \ \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B}$$

$$(\to E): \ \frac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}$$

$$(\wedge I): \ \frac{\Gamma \vdash M : A \quad \Gamma \vdash M : B}{\Gamma \vdash M : A \wedge B}$$

$$(\leq): \ \frac{\Gamma \vdash M : A}{\Gamma \vdash M : B} \ (A \leq B)$$

This system differs from that in [2] not just because of the syntax of types and of the definition of the $\leq$ relation, but also because of the restrictions (3.1(*iii*) on assumptions and of rule (ν).

If we allow assumptions with predicates which are not $\vee$-prime, then typing is not preserved under β_v-reduction. In fact, suppose that A, B and C are unrelated formulae (distinct propositional variables, say). Then we can deduce both $x : A \to C, y : A \wedge B \vdash xy : C$ and $x : B \to C, y : A \wedge B \vdash xy : C$, hence

$$y : A \wedge B \vdash \lambda x.xy : ((A \to C) \to C) \wedge ((B \to C) \to C) \sim [A \to C) \vee (B \to C] \to C,$$

by rules $(\to I), (\wedge I)$ and $(\leq)$. Suppose that we relax the restriction on the assumptions, and consider $z : (A \to C) \vee (B \to C)$. Then by rule $(\to E)$ and the admissibility of weakening we have:

$$z : (A \to C) \vee (B \to C), y : A \wedge B \vdash (\lambda x.xy)z : C.$$

Now $(\lambda x.xy)z$ β_v-reduces to zy, but $z : (A \to C) \vee (B \to C), y : A \wedge B \not\vdash zy : C$. This is an easy consequence of the Generation Lemma below (Lemma 3.4).

A second remark concerns rule (ν). Compared with [2], this rule is close to rule (ω):

$$(\omega): \ \frac{}{\Gamma \vdash M : \omega}$$

of [2], but for the restriction on the form of M. In fact, in the present system it is not true that any term has a type: a typical un-typeable term is $\Omega \equiv (\lambda x.xx)(\lambda x.xx)$. As a matter of fact, in our system only terms which are β_v-convertible to values have a type in the empty basis, hence (also) type ν. This is a basic choice: as terms will be discriminated according to the types we can assign to them, the fact that "divergent" closed terms (which are not β_v-convertible to a value) have no type will identify all of them.

In contrast, in [2] any term has a type, at least the "universal type" ω (trivially) and also type $\omega \to \omega$, because of the axiom $\omega \leq \omega \to \omega$. Had we introduced here the universal type ω, serious problems would immediately have arisen. Indeed, one would have, e.g., that $\vdash (\lambda xy.y)\Omega : A \to A$. But, once the inequation $\omega \leq \omega \to \omega$ has been dropped, we expect, as it happens in all standard models of call-by-value λ-calculus, that arrow types (in the empty basis) characterize terms evaluating to a value (see, e.g. [5]). But then $(\lambda xy.y)\Omega$ should be itself β_v-convertible with a value, which is not.

The next lemmas are routine, but essential for the technical development: the essence is that variables can be substituted by terms having the same type, and that, under certain conditions, the rules of the system can be reversed.

Lemma 3.2 (Syntactical properties)

i) (weakening) If $\Gamma \vdash M : A$ *then* $\Gamma, x : B \vdash M : A$.

ii) (thinning) If $\Gamma \vdash M : A$ *then* $\Gamma\lceil M \vdash M : A$ *, where* $\Gamma\lceil M = \{x : A \in \Gamma | x \in FV(M)\}$.

iii) (substitution) If $\Gamma, x : A \vdash M : B$ *and* $\Gamma \vdash N : A$ *then* $\Gamma \vdash M[N/x] : B$.

Proof: All statements above are proved by induction on derivations. In particular (iii) is proved by induction on the derivation of $\Gamma, x : A \vdash M : B$. We only observe that, if the last rule is (ν), then two cases are possible, according to the form of M:

(iii): Case $M \equiv y$: if $y \equiv x$ then $y[N/x] \equiv N$ and we have, using the hypothesis $\Gamma \vdash N : A$ and rule $(\leq)$, that $\Gamma \vdash N : \nu$; otherwise, if $y \not\equiv x$ then $y[N/x] \equiv y$, and $\Gamma \vdash y : \nu$ by rule (ν).

(iii): Case $M \equiv \lambda y.M'$: then $(\lambda y.M')[N/x] \equiv \lambda z.M'[y/z][N/x]$ (where z is a fresh variable) which is an abstraction; therefore $\Gamma \vdash M[N/x] : \nu$ is an instance of rule (ν).

When restricting to $\vee$-prime formulae, some useful properties concerning the type assignment system clearly reflect properties of natural deduction systems for propositional logic.

Lemma 3.3

i) The following rule is admissible:

$$(\leq_L): \ \frac{\Gamma, x : A \vdash M : B}{\Gamma, x : C \vdash M : B} \ (C \leq A \text{ and } C \text{ is } \vee\text{-prime})$$

ii) If $M : B$ *is derived from* $M : A_1 \ \cdots \ M : A_n$ *only by using rules* $(\wedge I)$ *and* $(\leq)$*, then* $A_1 \wedge \cdots \wedge A_n \leq B$.

Proof: (i) By induction on the derivation of $\Gamma, x : A \vdash M : B$.
(ii) By induction on derivations.

The Generation Lemma allows for a reverse reading of the rules. In particular, it establishes that, if the subject of the conclusion is an application or an abstraction, then, under suitable hypotheses, we have information on types which can be given to its immediate subterms.

Lemma 3.4 (Generation Lemma) *i) If* $\Gamma \vdash x : A$ *and* $A \not\sim \nu$ *then, for some* B, $x : B \in \Gamma$ *and* $B \leq A$.

ii) $\Gamma \vdash MN : A$ *iff* $\Gamma \vdash M : B \to A$ *and* $\Gamma \vdash N : B$ *for some* B.

iii) If $\Gamma \vdash \lambda x.M : A$ *and* $A \not\sim \nu$ *then* $\Gamma, x : B_i \vdash M : C_i$ *and* $\wedge_{i \in I}(B_i \to C_i) \leq A$ *for some* I, B_i, C_i.

iv) $\Gamma \vdash \lambda x.M : B \to C$ *and* B *is* $\vee$*-prime iff* $\Gamma, x : B \vdash M : C$.

Proof: *i*) Immediate, by Lemma 3.3 (ii).

ii) $(\Rightarrow)$ By induction on the derivation of $\Gamma \vdash MN : A$. The only non-trivial case is when the last rule applied is $(\wedge I)$, i.e. $A = A_1 \wedge A_2$. Then

$$\frac{\begin{array}{cc}\boxed{} & \boxed{} \\ \Gamma \vdash M : A_1 & \Gamma \vdash M : A_2\end{array}}{\Gamma \vdash M : A_1 \wedge A_2} \ (\wedge I)$$

is the last step. By induction, there are B_1, B_2 such that $\Gamma \vdash M : B_i \to A_i, \Gamma \vdash N : B_i$ for

$i = 1,2$. Then

$$\dfrac{\dfrac{\dfrac{\square}{\Gamma \vdash M : B_1 \to A_1}}{\Gamma \vdash M : B_1 \wedge B_2 \to A_1}(\leq) \qquad \dfrac{\dfrac{\square}{\Gamma \vdash M : B_2 \to A_2}}{\Gamma \vdash M : B_1 \wedge B_2 \to A_2}(\leq)}{\dfrac{\Gamma \vdash M : (B_1 \wedge B_2 \to A_1) \wedge (B_1 \wedge B_2 \to A_2)}{\Gamma \vdash M : B_1 \wedge B_2 \to A_1 \wedge A_2}(\leq)}(\wedge I)$$

and $\Gamma \vdash N : B_1 \wedge B_2$ by $(\wedge I)$.

$(\Leftarrow)$ By rule $(\to E)$.

iii) The derivation of $\Gamma \vdash \lambda x.M : A$ $(A \not\sim \nu)$ has the shape:

$$\dfrac{\dfrac{\dfrac{\square}{\Gamma, x : B_1 \vdash M : C_1}}{\Gamma \vdash \lambda x.M : B_1 \to C_1}(\to I) \quad \cdots \quad \dfrac{\dfrac{\square}{\Gamma, x : B_n \vdash M : C_n}}{\Gamma \vdash \lambda x.M : B_n \to C_n}(\to I) \quad \overline{\lambda x.M : \nu} \quad \cdots \quad \overline{\lambda x.M : \nu}}{\dfrac{\square}{\Gamma \vdash \lambda x.M : A}(\text{only } (\wedge I), (\leq))}$$

then by Lemma 3.3 (ii)

$$\wedge_{i \in I}(B_i \to C_i) \wedge \nu \quad \cdots \quad \wedge \nu \leq A.$$

$I \neq \emptyset$ because $A \not\sim \nu$ and $B_i \to C_i \leq \nu$ by (A1). Thus

$$\wedge_{i \in I}(B_i \to C_i) \leq A.$$

iv) In (ii) above, as $B \to C \not\sim \nu$ by Theorem 2.6 (iii), we can assume $A = B \to C$. Thus there exist I, B_i, C_i,

$$\Gamma, x : B_i \vdash M : C_i, \qquad \wedge_{i \in I}(B_i \to C_i) \leq B \to C.$$

Since B is $\vee$-prime, by Theorem 2.6 (iv),

$$\exists J \subseteq I, B \leq \wedge_{i \in J} B_i \text{ and } \wedge_{i \in J} C_i \leq C \;\Rightarrow\; \exists J \subseteq I \big[\forall i \in J, B \leq B_i \text{ and } \wedge_{i \in J} C_i \leq C\big].$$

Hence we have:

$$\dfrac{\dfrac{\dfrac{\Gamma, x : B_i \vdash M : C_i \text{ and } B \leq B_i \ (\forall i \in J) \quad \text{and } B \text{ is } \vee\text{-prime}}{\Gamma, x : B \vdash M : C_i \ (\forall i \in J)}(\wedge I)}{\Gamma, x : B \vdash M : \wedge_{i \in J} C_i}(\leq)}{\Gamma, x : B \vdash M : C}(\leq L)$$

The paper [1] considers a disjunction-elimination rule, which, as such, is not admissible in the present system. However, as in [5], a suitable restriction of it is admissible:

$$\dfrac{\Gamma \vdash M[V/x] : C}{\Gamma, x : A \vdash M : C \quad \Gamma, x : B \vdash M : C \quad \Gamma \vdash V : A \vee B}\ (V \text{ is a variable or an abstraction})$$

In fact, if $V \equiv y$ is a variable that does not occur in Γ, then $A \vee B \sim \nu$ (by Lemma 3.4 (i)); since ν is $\vee$-prime, both $A, B \sim \nu$, and the thesis follows form the Substitution Lemma (Lemma 3.2 (iii)). If $y : D$ is in Γ then D is $\vee$-prime and $D \leq A \vee B$: therefore, by rule $(\leq)$, either $\Gamma \vdash y : A$ or $\Gamma \vdash y : B$. In both cases, the thesis follows by substitution.

Suppose, instead, that $V \equiv \lambda y.M$. For $A \vee B \sim \nu$, we reason as in the previous case. Otherwise, if $A \vee B \not\sim \nu$, then, by Lemma 3.4 (iii), $\Gamma, y : D_i \vdash M : E_i$ for $i \in I$ (for some finite I and types D_i, E_i), such that $\wedge_{i \in I}(D_i \to E_i) \leq A \vee B$. By $(\to I)$ and $(\wedge I)$, $\Gamma \vdash \lambda y.M : \wedge_{i \in I}(D_i \to E_i)$; on the other hand, we know that $\wedge_{i \in I}(D_i \to E_i)$ is $\vee$-prime by Lemma 2.6 (v), hence either $\Gamma \vdash \lambda y.M : A$ or $\Gamma \vdash \lambda y.M : B$, and the result follows by substitution.

The restriction on the form of V is essential: notice that $x : C \to A \vee B, y : C \vdash xy : A \vee B$, but we cannot type, in the same basis, xy either by A or by B.

4 The call-by-value λ-calculus and its models

The type assignment system we have introduced does not respect the (unrestricted) β-rule, as it is the case, instead, for the system of [2]. More precisely, types are preserved neither under β-reduction, nor under β-expansion.

For example, one can deduce $(\lambda y.xyy)(uv) : C$ from the basis $\Gamma = \{x : (A \to A \to C) \wedge (B \to B \to C), u : D \to A \vee B, v : D\}$ as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash x : (A \to A \to C) \wedge (B \to B \to C)}{\Gamma \vdash x : A \to A \to C} \quad \Gamma \vdash y : A}{\Gamma \vdash xy : A \to C} \quad \Gamma \vdash y : A}{\Gamma \vdash xyy : C}}{\Gamma \vdash \lambda y.xyy : A \to C} \quad \boxed{}}{\Gamma \vdash \lambda y.xyy : (A \to C) \wedge (B \to C)} \quad \dfrac{\Gamma \vdash u : D \to A \vee B \quad \Gamma \vdash v : D}{\Gamma \vdash uv : A \vee B}}{\Gamma \vdash \lambda y.xyy : A \vee B \to C}}{\Gamma \vdash (\lambda y.xyy)(uv) : C}$$

One cannot deduce C from Γ for the normal form $x(uv)(uv)$. Also one can deduce $A \to A$ for $\lambda y.y$, but this type cannot be deduced for $(\lambda xy.y)((\lambda z.zz)(\lambda z.zz))$, which has no type at all.

Thus this type assignment system does not induce a λ-model. Instead it gives a model of the call-by-value λ-calculus. The call-by-value λ-calculus, as introduced by Plotkin [9], is obtained by restricting the β-rule to redexes whose argument is a value (i.e. a variable or an abstraction).

Definition 4.1 (Call-by-value λ-calculus) The set of *values* $Val \subset \Lambda$ is defined by

$$Val = Var \cup \{\lambda x.M | M \in \Lambda\}$$

where *Var* is the set of term variables. The call-by-value β-reduction rule is

$$(\beta_v) \quad (\lambda x.M)N \to_v M[N/x] \text{ if } N \in Val.$$

The contextual, reflexive, symmetric, and transitive closure of $\to_v$ is denoted by $=_v$.

Models of call-by-value λ-calculus are a generalisation of λ-models, in which a distinguished subset $\mathcal{V}$ of an applicative structure $\mathcal{D}$ is meant to interpret values: if one takes $\mathcal{V} = \mathcal{D}$, the following definition immediately coincides with the notion of syntactical λ-model of [7]. A slightly different definition can be found in [6]; the present one is from [10].

Definition 4.2 (Models of call-by-value λ-calculus) A *model* of call-by-value λ-calculus is a structure $\mathcal{M} = < \mathcal{D}, \mathcal{V}, \cdot, [\![\ ]\!]_\rho^{\mathcal{M}} >$, such that $\_ \cdot \_$ is a binary operation on $\mathcal{D}$, called *application* (i.e. $< \mathcal{D}, \cdot >$ is an applicative structure), $\mathcal{V} \subseteq \mathcal{D}$ and, for any environment $\rho : Var \to \mathcal{V}$, $[\![\ ]\!]_\rho^{\mathcal{M}}$ is a map from Λ to $\mathcal{D}$ satisfying (writing simply $[\![M]\!]_\rho$ for $[\![M]\!]_\rho^{\mathcal{M}}$):

i) $[\![x]\!]_\rho = \rho(x)$
ii) $[\![MN]\!]_\rho = [\![M]\!]_\rho \cdot [\![N]\!]_\rho$
iii) $[\![\lambda x.M]\!]_\rho \cdot d = [\![M]\!]_{\rho[x:=d]}$ if $d \in \mathcal{V}$
iv) $[\forall x \in FV(M), \rho(x) = \rho'(x) \Rightarrow [\![M]\!]_\rho = [\![M]\!]_{\rho'}$
v) $[\![\lambda x.M]\!]_\rho = [\![\lambda y.M[y/x]]\!]_\rho$ if $y \notin FV(M)$
vi) $[\forall d \in \mathcal{V}, [\![M]\!]_{\rho[x:=d]} = [\![N]\!]_{\rho[x:=d]}] \Rightarrow [\![\lambda x.M]\!]_\rho = [\![\lambda x.N]\!]_\rho$

vii) $M \in Val \Rightarrow [\![M]\!]_\rho \in \mathcal{V}$ for all ρ.

The soundness of this definition is proved in [10]: we state this fact in the next lemma.

Lemma 4.3 *If $M =_v N$ then $[\![M]\!]^{\mathcal{M}}_\rho = [\![N]\!]^{\mathcal{M}}_\rho$, for any model $\mathcal{M}$ and environment ρ.*

In analogy with [2], we will define a model using the set $\mathcal{F}$ of filters over L, generated by the $\mathsf{B+}$ implication $\leq$. The set of $\vee$-prime filters, as suggested by the remark at the end of the previous section, is meant as the set of values. There is, however, a slight mismatch between the present notion of filter and the standard one: we shall admit $\emptyset$ as an element of $\mathcal{F}$.

As a matter of fact, we shall construct a model of the call-by-value λ-calculus out of the set of filters: the denotation of a term M is a filter, namely the set of all types (formulae) that can be assigned to M. The fact that the empty set is considered as a filter is a consequence of our choice to use ν as the type of values, and of the fact that closed terms which are not β_v-convertible to a value have no type at all.

Definition 4.4 (FILTERS)

i) A *filter* over $\mathsf{B+}$ is a set $X \subseteq \mathsf{L}$ such that

- if $A \leq B$ and $A \in X$ then $B \in X$;
- if $A, B \in X$, then $A \wedge B \in X$.

ii) Let $\mathcal{F}$ be the set of all filters over $\mathsf{B+}$.

iii) A filter X over $\mathsf{B+}$ is a $\vee$-*prime filter* if $A \vee B \in X$ implies either $A \in X$ or $B \in X$.

iv) Let $\mathcal{PF}$ be the set of $\vee$-prime filters on $\mathsf{B+}$different from $\emptyset$.

v) If $X \subseteq \mathsf{L}$, $\uparrow X$ denotes the filter generated by X.

vi) $\uparrow A$ is short for $\uparrow \{A\}$.

$< \mathcal{F}, \subseteq >$ is a distributive lattice. As a domain it is algebraic, with finite (or compact) elements of the shape $\uparrow A$. Now we observe that $\uparrow B$ is $\vee$-prime if and only if B is $\vee$-prime; since any A is equivalent to a finite disjunction $A_1 \vee \cdots \vee A_k$ of $\vee$-prime formulae, we conclude that any finite element of $\mathcal{F}$ (namely any principal filter) factorizes into a finite intersection of $\vee$-prime filters: $\uparrow (A_1 \vee \cdots \vee A_k) = \uparrow A_1 \cap \cdots \cap \uparrow A_k$.

$\mathcal{F}$ is also a solution of the domain equation

$$\mathcal{D} \lhd [\mathcal{D} \to_\perp \mathcal{D}]_\perp$$

in the category of algebraic lattices, where $[\mathcal{D} \to_\perp \mathcal{D}]_\perp$ denotes the lifted space of strict continuous functions from $\mathcal{D}$ to $\mathcal{D}$. In fact $\emptyset$ and $\uparrow \nu$ are respectively the bottom of $\mathcal{F}$ and of $[\mathcal{F} \to_\perp \mathcal{F}]$. Moreover $\uparrow (A \to B)$ represents the strict step function $f_{\uparrow A, \uparrow B}$, where as usual

$$f_{\mathsf{a},\mathsf{b}}(\mathsf{d}) = \text{ if } \mathsf{a} \sqsubseteq \mathsf{d} \text{ then } \mathsf{b} \text{ else } \perp.$$

So, we can define the projection pair $\langle F, G \rangle$, where $F : \mathcal{F} \to [\mathcal{F} \to_\perp \mathcal{F}]_\perp$, $G : [\mathcal{F} \to_\perp \mathcal{F}]_\perp \to \mathcal{F}$, and $F \circ G = \mathsf{Id}_{[\mathcal{F} \to_\perp \mathcal{F}]_\perp}$, as follows:

$$F(X) = \bigsqcup \{ f_{\uparrow A, \uparrow B} \mid A \to B \in X \}$$

$$G(f) = \uparrow \{ A \to B \mid B \in f(\uparrow A) \} \cup \uparrow \nu.$$

Due to the presence of union types, we do not have an isomorphism between $\mathcal{F}$ and $[\mathcal{F} \to_\perp \mathcal{F}]_\perp$, since different filters represent the same function. For example, $\uparrow (A \to B) \vee (C \to D) \supseteq \uparrow A \wedge C \to B \vee D$ and the inclusion is proper, but $F(\uparrow (A \to B) \vee (C \to D)) = F(\uparrow$

$A \wedge C \to B \vee D) = f_{\uparrow A\wedge C, \uparrow B\vee D}$. As a matter of fact, there are solutions of the domain equation $\mathcal{D} \cong [\mathcal{D} \to_\perp \mathcal{D}]_\perp$ that can be described using intersection types only: see [6, 11, 10].

We will now define a notion of application on $\mathcal{F}$.

Definition 4.5 For $X, Y \in \mathcal{F}$, define

$$X \cdot Y = \{A \mid \exists B \in Y, B \to A \in X\}.$$

Observe that $X \cdot \emptyset = \emptyset \cdot Y = \emptyset$.

Lemma 4.6 *i)* $X \cdot Y \in \mathcal{F}$ *for all* $X, Y \in \mathcal{F}$.

ii) A non-empty filter X *is* $\vee$*-prime if and only if*

$$\forall B \in X\ \exists C \in X.\ C \leq B \ \&\ C \text{ is } \vee\text{-prime}.$$

Proof: *i*) First $X \cdot Y$ is upward closed: if $A \in X \cdot Y$ and $A \leq B$ then $C \to A \in X$, for some $C \in Y$. Then, by (R5), $C \to A \leq C \to B$, hence $C \to B \in X$, which is upward closed, so $B \in X \cdot Y$.

We now show that $X \cdot Y$ is closed under $\wedge$: let $A, B \in X \cdot Y$. By definition, there exist A', B' such that $A', B' \in Y$, $A' \to A, B' \to B \in X$. By (R4), $A' \to A \leq (A' \wedge B') \to A$, and, similarly, $B' \to B \leq (A' \wedge B') \to B$. By (A7), $((A' \wedge B') \to A) \wedge ((A' \wedge B') \to B) \leq (A' \wedge B') \to (A \wedge B) \in X$, since X is a filter. But $A' \wedge B' \in Y$, since Y is a filter too, so we conclude that $A \wedge B \in X \cdot Y$.

ii) ($\Leftarrow$) Let $A \vee B \in X$, then $C \leq A \vee B$, for some $\vee$-prime $C \in X$. Then immediately $C \leq A$ or $C \leq B$, so $A \in X$ or $B \in X$ since X is upward closed.
($\Rightarrow$) By Proposition A.3, $B \sim \mathbf{m}_{\vee\wedge}(B) = \bigvee_{i\in I} B_i$, where the B_i's are $\vee$-prime by Theorem 2.6 (ii). As X is $\vee$-prime, $B_i \in X$ for some $i \in I$, and clearly $B_i \leq B$.

The basic idea of the next definition is to interpret terms into $\mathcal{F}$ using the Leibnitzian principle for which objects are identified with the set of their properties (here formulae, or types).

Definition 4.7 (Term interpretation)

i) A basis Γ agrees with an environment $\rho : Var \to \mathcal{P}\mathcal{F}$ (notation $\Gamma \models \rho$) iff $x : B \in \Gamma$ implies $B \in \rho(x)$.

ii) The interpretation of λ-terms induced by $\vdash$ is defined by

$$[\![M]\!]^{\mathcal{F}}_\rho = \{A \in \mathsf{L} \mid \exists \Gamma \models \rho, \Gamma \vdash M : A\}.$$

The mapping $[\![M]\!]^{\mathcal{F}}_\rho$ is actually an interpretation from Λ to $\mathcal{F}$, as stated in the next Lemma.

Lemma 4.8 $[\![M]\!]^{\mathcal{F}}_\rho \in \mathcal{F}$, *for any* $M \in \Lambda$ *and environment* $\rho : Var \to \mathcal{P}\mathcal{F}$.

Proof: By rules $(\leq)$ and $(\wedge I)$.

The main fact we establish now about $\mathcal{F}$ is that, given the above definitions of application and interpretation, it is a (filter) model of the call-by-value λ-calculus.

Theorem 4.9 $\mathcal{M}_0 = < \mathcal{F}, \mathcal{P}\mathcal{F}, \cdot, [\![\ ]\!]^{\mathcal{F}}_\rho >$ *is a model of call-by-value* λ*-calculus.*

Proof: By checking that all conditions of Definition 4.2 are satisfied.

i) If $A \in [\![x]\!]^{\mathcal{F}}_\rho$, then $\Gamma \vdash x : A$ for some Γ such that $\Gamma \models \rho$. If $A \not\sim \nu$ (otherwise the thesis follows from the fact that $\rho(x) \neq \emptyset$) then $x : B \in \Gamma$ for some $B \leq A$, by Lemma 3.4 (i). This implies that $B \in \rho(x)$ and, therefore, also that $A \in \rho(x)$.

On the other hand, if $A \in \rho(x)$, then $\rho \models \{x : A\}$ and $x : A \vdash x : A$.

ii) Immediate by Lemma 3.4 (ii).

iii) Let $X \in \mathcal{PF}$, then

$$\begin{array}{lll} A \in [\![\lambda x.M]\!]^{\mathcal{F}}_{\rho} \cdot X & \Rightarrow & \exists B \in X.\, B \to A \in [\![\lambda x.M]\!]^{\mathcal{F}}_{\rho} \qquad \text{Def. 4.5} \\ & \Rightarrow & \exists C \in X.\, C \vee\text{-prime } \& \; C \to A \in [\![\lambda x.M]\!]^{\mathcal{F}}_{\rho} \qquad \text{Lem. 4.6 (ii), } C \leq B \\ & \Rightarrow & \exists C \in X, \Gamma.\, C \vee\text{-prime } \& \; \Gamma \models \rho \; \& \; \Gamma \vdash \lambda x.M : C \to A \\ & \Rightarrow & \exists C \in X, \Gamma.\, \Gamma, x : C \models \rho[x := X] \; \& \; \Gamma, x : C \vdash M : A \qquad \text{Lem. 3.4 (iv) } C \vee\text{-prime} \\ & \Rightarrow & A \in [\![M]\!]^{\mathcal{F}}_{\rho[x:=X]}. \end{array}$$

Vice versa

$$\begin{array}{lll} A \in [\![M]\!]^{\mathcal{F}}_{\rho[x:=X]} & \Rightarrow & \exists B, \Gamma.\, \Gamma, x : B \models \rho[x := X] \; \& \; \Gamma, x : B \vdash M : A \\ & \Rightarrow & \exists B \in X, \Gamma.\, \Gamma \models \rho \; \& \; \Gamma \vdash \lambda x.M : B \to A \\ & \Rightarrow & \exists B \in X.\, B \to A \in [\![\lambda x.M]\!]_{\rho} \\ & \Rightarrow & A \in [\![\lambda x.M]\!]_{\rho} \cdot X. \end{array}$$

iv) *v*), *vi*) Easy.

vii) If $M \equiv x \in Var$, then $[\![M]\!]^{\mathcal{F}}_{\rho} = \rho(x) \in \mathcal{PF}$, because ρ is a mapping from Var to $\mathcal{PF}$. Otherwise, suppose that $M \equiv \lambda x.M'$: then by Lemma 4.6 (ii), it suffices to prove that

$$B \in [\![\lambda x.M']\!]^{\mathcal{F}}_{\rho} \Rightarrow \exists C \in [\![\lambda x.M']\!]^{\mathcal{F}}_{\rho}.\, C \leq B \;\&\; C \text{ is } \vee\text{-prime}.$$

Assume $B \in [\![\lambda x.M']\!]^{\mathcal{F}}_{\rho}$, i.e. $\exists \Gamma \models \rho, \Gamma \vdash \lambda x.M' : B$. The case $B \sim \nu$ is trivial; let us suppose that $B \not\sim \nu$. By Lemma 3.4 (iii) we have that, for some I, B_i and C_i,

$$\forall i \in I.\, \Gamma, x : B_i \vdash M' : C_i \text{ and } \wedge_{i \in I} (B_i \to C_i) \leq B.$$

Then $\wedge_{i \in I}(B_i \to C_i) \in [\![\lambda x.M']\!]^{\mathcal{F}}_{\rho}$ by $(\to I)$ and $(\wedge I)$, where $\wedge_{i \in I}(B_i \to C_i)$ is $\vee$-prime by Theorem 2.6 (ii).

5 Conclusion

The fact that the system B+ gives naturally a type assignment system for the call-by-value λ-calculus is a pleasant surprise. Indeed, something more should be true about our system. First we strongly conjecture that an approximation theorem holds: given the right notion of approximant in the case of call-by-value λ-calculus (see e.g. [6]), we expect that the set of types that can be assigned to any term is exactly the set of all types that can be assigned to its approximate normal forms. Then some leading ideas of our construction, such as the fact that closed terms which can be typed by ν are exactly the convergent "programs", would have a clear and elegant proof.

There are, however, some open questions. First, we do not have a completeness theorem for the B+ based type assignment system along the lines of [2]. A deeper analysis of the correspondence between the filter model construction and the semantics of relevant logics is still on demand, and, we guess, should lead to a better understanding even of the results we presently have.

References

[1] F. Barbanera, M. Dezani-Ciancaglini and U. de'Liguoro, "Intersection and Union Types: Syntax and Semantics", *Information and Computation* **119** (1995), pp. 202-230.

[2] H. Barendregt, M. Coppo and M. Dezani-Ciancaglini, "A filter lambda model and the completeness of type assignment", *The journal of symbolic logic* **48** (1983), pp. 931-940.
[3] M. Coppo and M. Dezani-Ciancaglini, "An extension of the basic functionality theory for the λ-calculus", *Notre Dame journal of formal logic* **21** (1980), pp. 685-693.
[4] M. Dezani-Ciancaglini, U. de'Liguoro and A. Piperno, "Filter models for conjunctive-disjunctive λ-calculi", *Theoretical computer science* **170** (1996), pp. 83-128.
[5] M. Dezani-Ciancaglini, U. de'Liguoro and A. Piperno, "Fully abstract semantics for concurrent λ-calculus", *SIAM Journal on Computing* **27**, No. 5 (1998), 1376-1419.
[6] L. Egidi, F. Honsell and S. Ronchi della Rocca, "Operational, denotational and logical descriptions: A case study", *Fundamenta Informaticae* **16** (1992), pp. 149-169.
[7] R. Hindley and G. Longo, "Lambda-calculus models and extensionality", *Zeitschrift für Mathematische Logik* **26** (1980), pp.289-310.
[8] R. K. Meyer, "Types and the boolean system CB+", unpublished note, 1998.
[9] G. D. Plotkin, "Call-by-name, call-by-value and the λ-calculus", *Theoretical computer science* **1** (1975), pp. 125-159.
[10] A. Pravato, "Categorical models of untyped λ-calculi: a monoidial approach", Ph.D thesis, Università di Milano (1997).
[11] A. Pravato, S. Ronchi della Rocca and L. Roversi, "Categorical Semantics of the call-by-value Lambda Calculus", TACS '95, *Springer Lecture Notes in Computer Science* **902** (1995), pp. 381-396.
[12] R. Routley and R. K. Meyer, "The semantics of entailment III", *Journal of philosophical logic* **1** (1972), pp. 192-208.

Appendix A Properties of $\leq$

Specialisations of $\leq$ to the sets T_i are now introduced, whose definition exploits the syntactical form of the types in T_i.

Definition A.1 $\leq_i \subseteq T_i \times T_i$ $(i = \to, \vee, \wedge, \wedge\vee, \vee\wedge)$ are the least preorders such that

$(\leq_\to)$: $A \leq_\to B \Leftrightarrow$ either $A = B$ or $B = \nu$ or $A = A_1 \to A_2, B = B_1 \to B_2$ and $B_1 \leq_\wedge A_1, A_2 \leq_\vee B_2$

$(\leq_\vee)$: $\bigvee_{i\in I} A_i \leq_\vee \bigvee_{j\in J} B_j$ (where $A_i, B_j \in T_\to$) $\Leftrightarrow \forall i \in I \exists j \in J, A_i \leq_\to B_j$

$(\leq_\wedge)$: $\wedge_{i\in I} A_i \leq_\wedge \wedge_{j\in J} B_j$ (where $A_i, B_j \in T_\to$) $\Leftrightarrow \forall j \in J \exists i \in I, A_i \leq_\to B_j$

$(\leq_{\wedge\vee})$: $\wedge_{i\in I} A_i \leq_{\wedge\vee} \wedge_{j\in J} B_j$ (where $A_i, B_j \in T_\vee$) $\Leftrightarrow \forall j \in J \exists i \in I, A_i \leq_\vee B_j$

$(\leq_{\vee\wedge})$: $\bigvee_{i\in I} A_i \leq_{\vee\wedge} \bigvee_{j\in J} B_j$ (where $A_i, B_j \in T_\wedge$) $\Leftrightarrow \forall i \in I \exists j \in J, A_i \leq_\wedge B_j$.

Lemma A.2 $\leq_i$ $(i = \to, \vee, \wedge, \wedge\vee, \vee\wedge)$ *are reflexive and transitive.*

Proof: By induction on the definition of $\leq_i$.

The following proposition states that conjunctive/disjunctive normal forms are logically equivalent to their counterimages under $\mathbf{m}_{\wedge\vee}()$ and $\mathbf{m}_{\vee\wedge}()$, and that the specialised relations $\leq_i$ are actually restrictions of $\leq$ to the sets T_i respectively.

Property A.3 For all $A, B \in \mathsf{L}$ *:*

i) $A \sim \mathbf{m}_{\wedge\vee}(A) \sim \mathbf{m}_{\vee\wedge}(A)$.

ii) $A, B \in T_i, A \leq_i B \Rightarrow A \leq B$ *for* $i = \to, \vee, \wedge, \vee\wedge, \wedge\vee$.

iii) $A \leq B \Leftrightarrow \mathbf{m}_{\wedge\vee}(A) \leq_{\wedge\vee} \mathbf{m}_{\wedge\vee}(B) \Leftrightarrow \mathbf{m}_{\vee\wedge}(A) \leq_{\vee\wedge} \mathbf{m}_{\vee\wedge}(B)$.

Proof: *i*) By induction on A. E.g. if $A = B \to C$ then, by induction hypothesis, we have $B \sim \mathbf{m}_{\vee\wedge}(B) = \bigvee_{i\in I} B_i$ and $C \sim \mathbf{m}_{\wedge\vee}(C) = \wedge_{j\in J} C_j$, so that, by repeated uses of (A7), (A8), (R4) and (R5) we conclude that

$$B \to C \sim \bigvee_{i\in I} B_i \to \wedge_{j\in J} C_j \sim \wedge_{i\in I} \wedge_{j\in J} (B_i \to C_j) \sim \mathbf{m}_{\wedge\vee}(B \to C) = \mathbf{m}_{\vee\wedge}(B \to C).$$

ii) By straightforward induction on the definition of $\leq_i$.

iii) Implications ($\Leftarrow$) are immediate consequences of (i) and (ii). To prove ($\Rightarrow$) we use induction over $\leq$. All cases are simple calculations. E.g. case (R3) $A \leq B, C \leq D \Rightarrow A \vee C \leq B \vee D$: by induction hypothesis

$$\mathbf{m}_{\wedge\vee}(A) \leq_{\wedge\vee} \mathbf{m}_{\wedge\vee}(B) \Rightarrow \forall j \in J \exists i \in I \forall p \in I_i \exists q \in J_j, A_{ip} \leq_{\rightarrow} B_{jq},$$

where $\mathbf{m}_{\wedge\vee}(A) = \wedge_{i\in I} A_i$, $\mathbf{m}_{\vee\wedge}(A_i) = \bigvee_{p\in I_i} A_{ip}$, and $\mathbf{m}_{\wedge\vee}(B) = \wedge_{j\in J} B_j$, $\mathbf{m}_{\vee\wedge}(B_j) = \bigvee_{q\in J_j} B_{jq}$. Similarly,

$$\mathbf{m}_{\wedge\vee}(C) \leq_{\wedge\vee} \mathbf{m}_{\wedge\vee}(D) \Rightarrow \forall l \in L \exists k \in K \forall r \in K_k \exists s \in L_l, C_{kr} \leq_{\rightarrow} D_{ls},$$

where $\mathbf{m}_{\wedge\vee}(C) = \wedge_{k\in K} C_k$, $\mathbf{m}_{\vee\wedge}(C_k) = \bigvee_{k\in K_k} C_{kr}$ and $\mathbf{m}_{\wedge\vee}(D) = \wedge_{l\in L} D_l$, $\mathbf{m}_{\vee\wedge}(D_l) = \bigvee_{s\in L_l} D_{ls}$. Then we have

$$\begin{array}{ll} & \forall j \in J, l \in L \left[\exists i \in I \forall p \in I_i \exists q \in J_j, A_{ip} \leq_{\rightarrow} B_{jq} \text{ and } \exists k \in K \forall r \in K_k \exists s \in L_l, C_{kr} \leq_{\rightarrow} D_{ls}\right] \\ \Rightarrow & \forall j \in J, l \in L \exists i \in I, k \in K, \bigvee_{p\in I_i} A_{ip} \vee \bigvee_{r\in K_r} C_{kr} \leq_{\vee} \bigvee_{q\in J_j} B_{jq} \vee \bigvee_{s\in L_l} D_{ls} \\ \Rightarrow & \forall j \in J, l \in L \exists i \in I, k \in K, A_i \vee C_k \leq_{\vee} B_j \vee D_l \\ \Rightarrow & \wedge_{i\in I} \wedge_{k\in K} (A_i \vee C_k) \leq_{\wedge\vee} \wedge_{j\in J} \wedge_{l\in L} (B_j \vee D_l) \\ \Rightarrow & \mathbf{m}_{\wedge\vee}(A \vee C) \leq_{\wedge\vee} \mathbf{m}_{\wedge\vee}(B \vee D). \end{array}$$

We come eventually to the proof of Theorem 2.6.

Theorem A.4 (Properties of $\vee$-prime formulas)

i) ν and any propositional variable are $\vee$-prime.

ii) The formula $\wedge_{i\in I}(A_i \to B_i)$ is $\vee$-prime for all (finite) I and formulas A_i, B_i.

iii) If $\mathbf{m}_{\vee\wedge}(A) = \bigvee_{i\in I} A_i$ then A_i is $\vee$-prime, for all $i \in I$.

iv) $B \to C \not\sim \nu$ for all B, C.

v) $\wedge_{i\in I}(A_i \to B_i) \leq C \to D$ and C is $\vee$-prime imply $C \leq \wedge_{i\in J} A_i$ and $\wedge_{i\in J} B_i \leq D$ for some $J \subseteq I$.

Proof: *i*) Let $A \in \mathcal{PV} \cup \{\nu\}$, and suppose that $A \leq B \vee C$. Then $\mathbf{m}_{\vee\wedge}(A) = A$, while $\mathbf{m}_{\vee\wedge}(B \vee C) = \bigvee_{j\in J} B_j \vee \bigvee_{k\in K} C_k$, where $\mathbf{m}_{\vee\wedge}(B) = \bigvee_{j\in J} B_j$, and $\mathbf{m}_{\vee\wedge}(C) = \bigvee_{k\in K} C_k$. By Proposition A.3 (iii), we have $A \leq_{\vee\wedge} \bigvee_{j\in J} B_j \vee \bigvee_{k\in K} C_k$, so that, by Definition A.1, $A \leq_{\wedge} B_j$ or $A \leq_{\wedge} C_k$ for some j, k: then $A \leq B_j \leq B$ or $A \leq C_k \leq C$ by Proposition A.3 (iii).

ii) By Proposition A.3 (iii) we have:

$$\wedge_{i\in I}(A_i \to B_i) \leq C \vee D \Leftrightarrow \mathbf{m}_{\vee\wedge}(\wedge_{i\in I}(A_i \to B_i)) \leq_{\vee\wedge} \mathbf{m}_{\vee\wedge}(C \vee D).$$

Now $\mathbf{m}_{\vee\wedge}(\wedge_{i\in I}(A_i \to B_i))$ is a conjunction of arrows, namely a formula with no disjunction at the top level; on the other hand $\mathbf{m}_{\vee\wedge}(C \vee D)$ has the form $\bigvee_{k\in K} C_k \vee \bigvee_{l\in L} D_l$. By definition of $\leq_{\vee\wedge}$ we immediately have that $\mathbf{m}_{\vee\wedge}(\wedge_{i\in I}(A_i \to B_i)) \leq_{\wedge} C_k$ or $\mathbf{m}_{\vee\wedge}(\wedge_{i\in I}(A_i \to B_i)) \leq_{\wedge} D_l$, for some k, l; therefore the thesis follows by Proposition A.3 (i) and (ii).

iii) Parts (i) and (ii) imply that any formula in $T_{\wedge}$ is $\vee$-prime: hence the thesis follows by the definition of $\mathbf{m}_{\vee\wedge}()$.

iv) By contra-position. Suppose that $\nu \leq B \to C$: then, by Proposition A.3 (iii), $\nu \leq_{\wedge\vee} \mathbf{m}_{\wedge\vee}(B \to C) = \wedge_{i\in I, j\in J}(B_i \to C_j)$. This implies that there exist i and j such that $\nu \leq_{\rightarrow} B_i \to C_j$, which is not.

v) Let first compute:

$$\mathbf{m}_{\vee\wedge}(\wedge_{i\in I}(A_i \to B_i)) = \wedge_{i\in I} \left[\wedge_{h\in H_i} \wedge_{l\in L_i} (A_{i,h} \to B_{i,l})\right],$$

where $\mathbf{m}_{\vee\wedge}(A_i) = \bigvee_{h\in H_i} A_{i,h}$, and $\mathbf{m}_{\wedge\vee}(B_i) = \wedge_{l\in L_i} B_{i,l}$. On the other hand suppose that $\mathbf{m}_{\vee\wedge}(C \to D) = \wedge_{p\in P} \wedge_{q\in Q} (C_p \to D_q)$, where $\mathbf{m}_{\vee\wedge}(C) = \bigvee_{p\in P} C_p$, and $\mathbf{m}_{\wedge\vee}(D) = \wedge_{q\in Q} D_q$. By Proposition A.3 (iii) and the definition of $\leq_{\wedge\vee}$ we have

$$\forall p \in P, q \in Q\ \exists i \in I, h \in H_i, l \in l_i.\ C_p \leq_\wedge A_{i,h}\ \&\ B_{i,l} \leq_\vee D_q.$$

By Proposition A.3 (i), $C \sim \bigvee_{p\in P} C_p$: hence, since C is $\vee$-prime, there exists $p \in P$ such that $C \leq C_p$. Choose one such p and, for any $q \in Q$, define

$$J_q = \{i \mid \exists i \in I, h \in H_i, l \in l_i.\ C_p \leq_\wedge A_{i,h}\ \&\ B_{i,l} \leq_\vee D_q\},$$

which is non-empty by the above statement. Finally, we take $J = \bigcup_{q\in Q} J_q$. Now, for all $i \in J$, there exists $i \in H_i$ such that $C_p \leq A_{i,h} \leq A_i$: therefore $C \leq C_p \leq \wedge_{i\in J} A_i$.

To conclude, for all $q \in Q$ there is $i \in J_q$ and $l \in L_i$ such that $B_i \leq B_{i,l} \leq D_q$: then $\wedge_{i\in J} B_i \leq D_q$ for all q, and, therefore, $\wedge_{i\in J} B_i \leq \wedge_{q\in Q} D_q \sim D$.