Chapter 5
A Survey of Annotated Logics

Seiki Akama

Dedicated to Jair Minoro Abe for his 60th birthday

Abstract Annotated logics have been originally developed as foundations for paraconsistent logic programming, and later developed as paracomplete and paraconsistent logics by J.M. Abe and others. In this paper, we present the formalization of propositional and predicate annotated logics. We also review some formal issues.

Keywords Paraconsistent logics · Annotated logics · Paraconsistency · Paracompleteness · Paraconsistent logic programming

5.1 Introduction

One of J.M. Abe's contributions to paraconsistent logics is to establish the so-called *annotated logics*, which are paraconsistent and in general paracomplete. They have been developed as theoretical foundations for paraconsistent logic programming for inconsistent knowledge; see Subrahmanian [45] and Blair and Subrahmanian [22]. Later, they have been studied as the systems of paraconsistent logic by many people; see [1, 26, 30].

Abe explored many applications of annotated logics to various areas, including engineering. It is thus interesting to sketch the basics of annotated logics. We show their formal aspects without proofs. The complete exposion of annotated logics can be found in Abe et al. [8].

S. Akama (✉)
C-Republic, 1-20-1 Higashi-Yurigaoka, Asao-ku, Kawasaki 215-0012, Japan
e-mail: akama@jcom.home.ne.jp

S. Akama (ed.), *Towards Paraconsistent Engineering*, Intelligent Systems
Reference Library 110, DOI 10.1007/978-3-319-40418-9_5

The chapter is structured as follows. In Sect. 5.2, we present propositional annotated logics $P\tau$. In Sect. 5.3, we describe predicate annotated logics $Q\tau$. Section 5.4 gives Curry algebras as an algebraic semantics for annotated logics. We give some conclusions in Sect. 5.5.

5.2 Propositional Annotated Logics $P\tau$

As reviewed in Chap. 2, paraconsistent logics have been developed as the basis to formalize inconsistent but non-trivial theories, and many systems of paraconsistent logic have been proposed in the literature. Recently, we can find several interesting applications of paraconsistent logics for various areas including mathematics, philosophy and computer science.

There are historically three important systems of paraconsistent logic; see Priest et al. [40]. Jaśkowski proposed a paraconsistent propositional logic called *discursive logic* (or discussive logic) in 1948; see Jaśkowski [33, 34]. Discursive logic is a kind of modal approach to paraconsistency.

Da Costa proposed the so-called *C-system*, which is based on the non-standard interpretation of negation which is dual to intuitionistic negation. He developed propositional and predicate logic for C-system.

Relevance logic (or relevant logic) due to Anderson and Belnap formalizes a correct interpretation of implication, and some of relevant systems can be viewed as paraconsistent; see Anderson and Belnap [15] and Anderson et al. [16] and Routley et al. [44]. For a comprehensive survey, consult Dunn [31].

Since then, a lot of work has been done to develop a paraconsistent logic from some motivation. For a recent survey of paraconsistent logic, see Priest [42].

In 1979, Priest [41] proposed a *logic of paradox*, denoted LP, to deal with the semantic paradox.

Batens developed the so-called *adaptative logics* in Batens [18, 19] as improvements of *dynamic dialectical logics* developed in Batens [17]. *Inconsistency-adaptive logics* as developed by Batens [18] can be regarded as paraconsistent and non-monotonic logics.

Carnelli's *Logics of Formal Inconsistency* (LFI) are logical systems that deal with consistency and inconsistency as object-level concept; see Carnelli et al. [23]. And several paraconsistent systems can be interpreted in LFI.

Now, we turn to a formal presentation of annotated logics. Before doing it, we introduce some basic concepts. Let T be a theory whose underlying logic is L. T is called *inconsistent* when it contains theorems of the form A and $\neg A$ (the negation of A), i.e.,

$$T \vdash_L A \text{ and } T \vdash_L \neg A$$

where $\vdash_L$ denotes the provability relation in L. If T is not inconsistent, it is called *consistent*.

T is said to be *trivial*, if all formulas of the language are also theorems of T. Otherwise, T is called *non-trivial*. Then, for trivial theory T, $T \vdash_L B$ for any formula B. Note that trivial theory is not interesting since every formula is provable.

If L is classical logic (or one of several others, such as intuitionistic logic), the notions of inconsistency and triviality agree in the sense that T is inconsistent iff T is trivial. So, in trivial theories the extensions of the concepts of formula and theorem coincide.

A *paraconsistent logic* is a logic that can be used as the basis for inconsistent but non-trivial theories. In this regard, paraconsistent theories do not satisfy, in general, the *principle of non-contradiction*, i.e., $\neg(A \wedge \neg A)$.

We can also define a paracomplete logic. A *paracomplete logic* is a logic, in which the *principle of excluded middle*, i.e., $A \vee \neg A$ is not a theorem. In this sense, intuitionistic logic is one of the paracomplete logics. A *paracomplete theory* is a theory based on paracomplete logic.

Finally, the logic which is simultaneously paraconsistent and paracomplete is called *non-alethic logic*.

The important problems handled by paraconsistent logics include the paradoxes of set theory, the semantic paradoxes, and some issues in dialectics. These problems are central to philosophy and philosophical logic. However, paraconsistent logics have later found interesting applications in AI, in particular, expert systems, belief, and knowledge, among others, since the 1980s; see da Costa and Subrahmanian [29].

Annotated logics were introduced by Subrahmanian to provide a foundation for paraconsistent logic programming; see Subrahmanian [45] and Blair and Subrahmanian [22]. Paraconsistent logic programming can be seen as an extension of logic programming based on classical logic.

In 1989, Kifer and Lozinskii proposed a logic for reasoning with inconsistency, which is related to annotated logics; see Kifer and Lozinskii [35, 36]. In the same year, Kifer and Subrahmanian extended annotated logics by introducing *generalized annotated logics* in the context of logic programming; see Kifer and Subrahmanian [37]. In 1990, a resolution-style automatic theorem-proving method for annotated logics was implemented; see da Costa et al. [28].

Of course, annotated logics were developed as a foundation for paraconsistent logic programming, but they have interesting features to be examined by logicians. Formally, annotated logics are ingeneral non-alethic in the sense of the above terminology. From a viewpoint of paraconsistent logicians, annotated logics were regarded as new systems.

In 1991, da Costa and others started to study annotated logics from a foundational point of view; see da Costa et al. [26, 30]. In these works, propositional and predicate annotated logics were formally investigated by presenting axiomatization, semantics and completeness results, and some applications of annotated logics were briefly surveyed.

In 1992, Jair Minoro Abe wrote Ph.D. thesis on the foundations of annotated logics under Prof. Newton C.A. da Costa at University of São Paulo; see Abe [1]. Abe proposed annotated modal logics which extend annotated logics with modal operator in Abe [2]; also see Akama and Abe [9].

Some formal results including decidability annotated logics were presented in Abe and Akama [6]. Abe and Akama also investigated predicate annotated logics by the method of ultraproducts in Abe and Akama [5]. Abe [3] studied an algebraic semantics of annotated logics.

Now, we formally introduce annotated logics. The language of the propositional annotated logics $P\tau$. We denote by L the language of $P\tau$. Annotated logics are based on some arbitrary fixed finite lattice called a *lattice of truth-values* denoted by $\tau = \langle |\tau|, \leq, \sim \rangle$, which is the complete lattice with the ordering $\leq$ and the operator $\sim : |\tau| \rightarrow |\tau|$.

Here, $\sim$ gives the "meaning" of atomic-level negation of $P\tau$. We also assume that $\top$ is the top element and $\bot$ is the bottom element, respectively. In addition, we use two lattice-theoretic operations: $\vee$ for the least upper bound and $\wedge$ for the greatest lower bound.[1]

Definition 5.1 (*Symbols*) The symbols of $P\tau$ are defined as follows:

1. Propositional symbols: $p, q, \ldots$ (possibly with subscript)
2. Annotated constants: $\mu, \lambda, \ldots \in |\tau|$
3. Logical connectives: $\wedge$ (conjunction), $\vee$ (disjunction), $\rightarrow$ (implication), and $\neg$ (negation)
4. Parentheses: (and)

Definition 5.2 (*Formulas*) Formulas are defined as follows:

1. If p is a propositional symbol and $\mu \in |\tau|$ is an annotated constant, then p_μ is a formula called an *annotated atom*.
2. If F is a formula, then $\neg F$ is a formula.
3. If F and G are formulas, then $F \wedge G$, $F \vee G$, $F \rightarrow G$ are formulas.
4. If p is a propositional symbol and $\mu \in |\tau|$ is an annotated constant, then a formula of the form $\neg^k p_\mu$ $(k \geq 0)$ is called a *hyper-literal*. A formula which is not a hyper-literal is called a *complex formula*.

Here, some remarks are in order. The annotation is attached only at the atomic level. An annotated atom of the form p_μ can be read "it is believed that p's truth-value is at least μ". In this sense, annotated logics incorporate the feature of many-valued logics.

A hyper-literal is special kind of formula in annotated logics. In the hyper-literal of the form $\neg^k p_\mu$, $\neg^k$ denotes the k's repetition of $\neg$. More formally, if A is an annotated atom, then $\neg^0 A$ is A, $\neg^1 A$ is $\neg A$, and $\neg^k A$ is $\neg(\neg^{k-1} A)$. The convention is also use for $\sim$.

Next, we define some abbreviations.

[1]We employ the same symbols for lattice-theoretical operations as the corresponding logical connectives.

Definition 5.3 Let A and B be formulas. Then, we put:

$$A \leftrightarrow B =_{def} (A \rightarrow B) \wedge (B \rightarrow A)$$
$$\neg_* A =_{def} A \rightarrow (A \rightarrow A) \wedge \neg(A \rightarrow A)$$

Here, $\leftrightarrow$ is called the *equivalence* and $\neg_*$ *strong negation*, respectively.

Observe that strong negation in annotated logics behaves classically in that it has all the properties of classical negation.

We turn to a semantics for $P\tau$. We here describe a *model-theoretic semantics* for $P\tau$. Let **P** is the set of propositional variables. An *interpretation* I is a function $I : \mathbf{P} \rightarrow \tau$. To each interpretation I, we associate a *valuation* $v_I : \mathbf{F} \rightarrow \mathbf{2}$, where **F** is a set of all formulas and $\mathbf{2} = \{0, 1\}$ is the set of truth-values. Henceforth, the subscript is suppressed when the context is clear.

Definition 5.4 (*Valuation*) A valuation v is defined as follows:
If p_λ is an annotated atom, then

$v(p_\lambda) = 1$ iff $I(p) \geq \lambda$,
$v(p_\lambda) = 0$ otherwise,
$v(\neg^k p_\lambda) = v(\neg^{k-1} p_{\sim\lambda})$, where $k \geq 1$.

If A and B are formulas, then

$v(A \wedge B) = 1$ iff $v(A) = v(B) = 1$,
$v(A \vee B) = 0$ iff $v(A) = v(B) = 0$,
$v(A \rightarrow B) = 0$ iff $v(A) = 1$ and $v(B) = 0$.

If A is a complex formula, then

$v(\neg A) = 1 - v(A)$.

Say that the valuation v *satisfies* the formula A if $v(A) = 1$ and that v *falsifies* A if $v(A) = 0$. For the valuation v, we can obtain the following lemmas.

Lemma 5.1 *Let p be a propositional variable and $\mu \in |\tau|$ $(k \geq 0)$, then we have:*

$v(\neg^k p_\mu) = v(p_{\sim^k \mu})$.

Lemma 5.2 *Let p be a propositional variable, then we have:*

$v(p_\perp) = 1$

Lemma 5.3 *For any complex formula A and B and any formula F, the valuation v satisfies the following:*

1. $v(A \leftrightarrow B) = 1$ iff $v(A) = v(B)$
2. $v((A \rightarrow A) \wedge \neg(A \rightarrow A)) = 0$
3. $v(\neg_* A) = 1 - v(A)$
4. $v(\neg F \leftrightarrow \neg_* F) = 1$

We here define the notion of semantic consequence relation denoted by $\models$. Let Γ be a set of formulas and F be a formula. Then, F is a *semantic consequence* of Γ, written $\Gamma \models F$, iff for every v such that $v(A) = 1$ for each $A \in \Gamma$, it is the case that $v(F) = 1$. If $v(A) = 1$ for each $A \in \Gamma$, then v is called a *model* of Γ. If Γ is empty, then $\Gamma \models F$ is simply written as $\models F$ to mean that F is *valid*.

Lemma 5.4 *Let p be a propositional variable and $\mu, \lambda \in \mid \tau \mid$. Then, we have:*

1. $\models p_\perp$
2. $\models p_\mu \to p_\lambda,\ \mu \geq \lambda$
3. $\models \neg^k p_\mu \leftrightarrow p_{\sim^k \mu},\ k \geq 0$

The consequence relation $\models$ satisfies the next property.

Lemma 5.5 *Let A, B be formulas. Then, if $\models A$ and $\models A \to B$ then $\models B$.*

Lemma 5.6 *Let F be a formula and p a propositional variable. $(\mu_i)_{i \in J}$ be an annotated constant, where J is an indexed set. Then, if $\models F \to p_\mu$, then $F \to p_{\mu_i}$, where $\mu = \bigvee \mu_i$.*

As a corollary to Lemma 5.6, we can obtain the following lemma.

Lemma 5.7 $\models p_{\lambda_1} \wedge p_{\lambda_2} \wedge \cdots \wedge p_{\lambda_m} \to p_\lambda$, *where* $\lambda = \bigvee_{i=1}^{m} \lambda_i$.

Next, we discuss some results related to paraconsistency and paracompleteness.

Definition 5.5 (*Complementary property*) A truth-value $\mu \in \tau$ has the *complementary property* if there is a λ such that $\lambda \leq \mu$ and $\sim \lambda \leq \mu$. A set $\tau' \subseteq \tau$ has the *complementary property* iff there is some $\mu \in \tau'$ such that μ has the complementary property.

Definition 5.6 (*Range*) Suppose I is an interpretation of the language L. The *range* of I, denoted $range(I)$, is defined to be $range(I) = \{\mu \mid (\exists A \in B_L) I(A) = \mu\}$, where B_L denotes the set of all ground atoms in L.

For $P\tau$, ground atoms correspond to propositional variables. If the range of the interpretation I satisfies the complementary property, then the following theorem can be established.

Theorem 5.1 *Let I be an interpretation such that $range(I)$ has the complementary property. Then, there is a propositional variable p and $\mu \in |\tau|$ such that*

$$v(p_\mu) = v(\neg p_\mu) = 1.$$

Theorem 5.1 states that there is a case in which for some propositional variable it is both true and false, i.e., inconsistent. The fact is closely tied with the notion of paraconsistency.

Definition 5.7 (¬-*inconsistency*) We say that an interpretation I is ¬-*inconsistent* iff there is a propositional variable p and an annotated constant $\mu \in |\tau|$ such that $v(p_\mu) = v(\neg p_\mu) = 1$.

Therefore, ¬-inconsistency means that both A and $\neg A$ are simultaneously true for some atomic A. Below, we formally define the concepts of non-triviality, paraconsistency and paracompleteness.

Definition 5.8 (*Non-triviality*) We say that an interpretation I is *non-trivial* iff there is a propositional variable p and an annotated constant $\mu \in |\tau|$ such that $v(p_\mu) = 0$.

By Definition 5.8, we mean that not every atom is valid if an interpretation is non-trivial.

Definition 5.9 (*Paraconsistency*) We say that a interpretation I is *paraconsistent* iff it is both ¬-inconsistent and non-trivial. $P\tau$ is called *paraconsistent* iff there is an interpretation of I of $P\tau$ such that I is paraconsistent.

Definition 5.9 allows the case in which both A an $\neg A$ are true, but some formula B is false in some paraconsistent interpretation I.

Definition 5.10 (*Paracompleteness*) We say that an interpretation I is *paracomplete* iff there is a propositional variable p and a annotated constant $\lambda \in |\tau|$ such that $v(p_\lambda) = v(\neg p_\lambda) = 0$. $P\tau$ is called *paracomplete* iff there is an interpretation I of $P\tau$ such that I is paracomplete.

From Definition 5.10, we can see that in the paracomplete interpretation I, both A and $\neg A$ are false. We say that $P\tau$ is *non-alethic* iff it is both paraconsistent and paracomplete. Intuitively speaking, paraconsistent logic can deal with inconsistent information and paracomplete logic can handle incomplete information.

This means that non-alethic logics like annotated logics can serve as logics for expressing both inconsistent and incomplete information. This is one of the starting points of our study of annotated logics.

As the following Theorems 5.2 and 5.3 indicate, paraconsistency and paracompleteness in $P\tau$ depend on the cardinality of τ.

Theorem 5.2 *$P\tau$ is paraconsistent iff $card(\tau) \geq 2$, where $card(\tau)$ denotes the cardinality (cardinal number) of the set τ.*

Theorem 5.3 *If $card(\tau) \geq 2$, then there are annotated systems $P\tau$ such that they are paracomplete.*

The above two theorems imply that to formalize a non-alethic logic based on annotated logics we need at least both the top and bottom elements of truth-values. The simplest lattice of truth-values is $FOUR$ in Belnap [20, 21].

Definition 5.11 (*Theory*) Given an interpretation I, we can define the theory $Th(I)$ associated with I to be a set:

$$Th(I) = Cn(\{p_\mu \mid p \in \mathbf{P} \text{ and } I(p) \geq \mu\}).$$

Here, Cn is the semantic consequence relation, i.e.,

$$Cn(\Gamma) = \{F \mid F \in \mathbf{F} \text{ and } \Gamma \models F\}.$$

Here, Γ is a set of formulas.

$Th(I)$ can be extended for any set of formulas.

Theorem 5.4 *An interpretation I is $\neg$-inconsistent iff $Th(\Gamma)$ is $\neg$-inconsistent.*

Theorem 5.5 *An interpretation I is paraconsistent iff $Th(I)$ is paraconsistent.*

The next lemma states that the replacement of equivalent formulas within the scope of $\neg$ does not hold in $P\tau$ as in other paraconsistent logics.

Lemma 5.8 *Let A be any hyper-literal. Then, we have:*

1. $\models A \leftrightarrow ((A \to A) \to A)$
2. $\not\models \neg A \leftrightarrow \neg(((A \to A) \to A))$
3. $\models A \leftrightarrow (A \wedge A)$
4. $\not\models \neg A \leftrightarrow \neg(A \wedge A)$
5. $\models A \leftrightarrow (A \vee A)$
6. $\not\models \neg A \leftrightarrow \neg(A \vee A)$

As obvious from the above proofs, (1), (3) and (5) hold for any formula A. But, (2), (4) and (6) cannot be generalized for any A.

By the next theorem, we can find the connection of $P\tau$ and the positive fragment of classical propositional logic C.

Theorem 5.6 *If $F_1, \ldots, F_n$ are complex formulas and $K(A_1, \ldots, A_n)$ is a tautology of C, where $A_1, \ldots, A_n$ are the sole propositional variable occurring in the tautology, then $K(F_1, \ldots, F_n)$ is valid in $P\tau$. Here, $K(F_1, \ldots, F_n)$ is obtained by replacing each occurrence of A_i, $1 \leq i \leq n$, in K by F_i.*

Next, we consider the properties of strong negation $\neg_*$.

Theorem 5.7 *Let A, B be any formulas. Then,*

1. $\models (A \to B) \to ((A \to \neg_* B) \to \neg_* A)$
2. $\models A \to (\neg_* A \to B)$
3. $\models A \vee \neg_* A$

Theorem 5.7 tells us that strong negation has all the basic properties of classical negation. Namely, (1) is a principle of *reductio ad abusurdum*, (2) is the related principle of the law of non-contradiction, and (3) is the law of excluded middle. Note that $\neg$ does not satisfy these properties. It is also noticed that for any complex formula $A \models \neg A \leftrightarrow \neg_* A$ but that for any hyper-literal $Q \not\models \neg Q \leftrightarrow \neg_* Q$.

From these observations, $P\tau$ is a paraconsistent and paracomplete logic, but adding strong negation enables us to perform classical reasoning.

Next, we provide an axiomatization of $P\tau$ in the Hilbert style. There are many ways to axiomatize a logical system, one of which is the *Hilbert system*. Hilbert system can be defined by the set of *axioms* and *rules of inference*. Here, an axiom is a formula to be postulated as valid, and rules of inference specify how to prove a formula.

We are now ready to give a Hilbert style axiomatization of $P\tau$, called $\mathcal{A}\tau$. Let A, B, C be arbitrary formulas, F, G be complex formulas, p be a propositional variable, and λ, μ, λ_i be annotated constant. Then, the postulates are as follows (cf. Abe [1]):

Postulates for $\mathcal{A}\tau$

($\rightarrow_1$) $(A \rightarrow (B \rightarrow A)$
($\rightarrow_2$) $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$
($\rightarrow_3$) $((A \rightarrow B) \rightarrow A) \rightarrow A$
($\rightarrow_4$) $A, A \rightarrow B/B$
($\wedge_1$) $(A \wedge B) \rightarrow A$
($\wedge_2$) $(A \wedge B) \rightarrow B$
($\wedge_3$) $A \rightarrow (B \rightarrow (A \wedge B))$
($\vee_1$) $A \rightarrow (A \vee B)$
($\vee_2$) $B \rightarrow (A \vee B)$
($\vee_3$) $(A \rightarrow C) \rightarrow ((B \rightarrow C) \rightarrow ((A \vee B) \rightarrow C))$
($\neg_1$) $(F \rightarrow G) \rightarrow ((F \rightarrow \neg G) \rightarrow \neg F)$
($\neg_2$) $F \rightarrow (\neg F \rightarrow A)$
($\neg_3$) $F \vee \neg F$
(τ_1) $p_\perp$
(τ_2) $\neg^k p_\lambda \leftrightarrow \neg^{k-1} p_{\sim\lambda}$
(τ_3) $p_\lambda \rightarrow p_\mu$, where $\lambda \geq \mu$
(τ_4) $p_{\lambda_1} \wedge p_{\lambda_2} \wedge \cdots \wedge p_{\lambda_m} \rightarrow p_\lambda$, where $\lambda = \bigvee_{i=1}^{m} \lambda_i$

Here, except ($\rightarrow_4$), these postulates are axioms. ($\rightarrow_4$) is a rule of inferences called *modus ponens* (MP).

In da Costa et al. [30], a different axiomatization is given, but it is essentially the same as ours. There, the postulates for implication are different. Namely, although ($\rightarrow_1$) and ($\rightarrow_3$) are the same (although the naming differs), the remaining axiom is:

$$(A \rightarrow B) \rightarrow ((A \rightarrow (B \rightarrow C)) \rightarrow (A \rightarrow C))$$

It is well known that there are many ways to axiomatize the implicational fragment of classical logic C. In the absence of negation, we need the so-called *Pierce's law* ($\rightarrow_3$) for C.

In ($\neg_1$), ($\neg_2$), ($\neg_3$), F and G are complex formulas. In general, without this restriction on F and G, these are not sound rules due to the fact that they are not admitted in annotated logics.

da Costa et al. [30] fuses (τ_1) and (τ_2) as the single axiom in conjunctive form. But, we separate it in two axioms for our purposes. Also there is a difference in the final axiom. They present it for infinite lattices as

$$A \rightarrow p_{\lambda_j} \text{ for every } j \in J, \text{ then } A \rightarrow p_\lambda, \text{ where } \lambda = \bigvee_{j \in J} \lambda_j.$$

If τ is a finite lattice, this is equivalent to the form of (τ_2).

As usual, we can define a *syntactic consequence relation* in $P\tau$. Let Γ be a set of formulas and G be a formula. Then, G is a syntactic consequence of Γ, written $\Gamma \vdash G$, iff there is a finite sequence of formulas $F_1, F_2, \ldots, F_n$, where F_i belongs to Γ, or F_i is an axiom ($1 \leq i \leq n$), or F_j is an immediate consequence of the previous two formulas by ($\rightarrow_4$). This definition can extend for the transfinite case in which n is an ordinal number. If $\Gamma = \emptyset$, i.e. $\vdash G$, G is a *theorem* of $P\tau$.

Let Γ, Δ be sets of formulas and A, B be formulas. Then, the consequence relation $\vdash$ satisfies the following conditions.

1. if $\Gamma \vdash A$ and $\Gamma \subset \Delta$ then $\Delta \vdash A$.
2. if $\Gamma \vdash A$ and $\Delta, A \vdash B$ then $\Gamma, \Delta \vdash B$.
3. if $\Gamma \vdash A$, then there is a finite subset $\Delta \subset \Gamma$ such that $\Delta \vdash A$.

In the Hilbert system above, the so-called *deduction theorem* holds.

Theorem 5.8 (Deduction theorem) *Let Γ be a set of formulas and A, B be formulas. Then, we have:*

$$\Gamma, A \vdash B \;\Rightarrow\; \Gamma \vdash A \rightarrow B.$$

The following theorem shows some theorems related to strong negation.

Theorem 5.9 *Let A and B be any formula. Then,*

1. $\vdash A \vee \neg_* A$
2. $\vdash A \rightarrow (\neg_* A \rightarrow B)$
3. $\vdash (A \rightarrow B) \rightarrow ((A \rightarrow \neg_* B) \rightarrow \neg_* A)$

From Theorems 5.9, 5.10 follows.

Theorem 5.10 *For arbitrary formulas A and B, the following hold:*

1. $\vdash \neg_*(A \wedge \neg_* A)$
2. $\vdash A \leftrightarrow \neg_*\neg_* A$
3. $\vdash (A \wedge B) \leftrightarrow \neg_*(\neg_* A \vee \neg_* B)$
4. $\vdash (A \to B) \leftrightarrow (\neg_* A \vee B)$
5. $\vdash (A \vee B) \leftrightarrow \neg_*(\neg_* A \wedge \neg_* B)$

Theorem 5.10 implies that by using strong negation and a logical connective other logical connectives can be defined as in classical logic. If $\tau = \{t, f\}$, with its operations appropriately defined, we can obtain classical propositional logic in which $\neg_*$ is classical negation.

Now, we provide some formal results of $P\tau$ including completeness and decidability.

Lemma 5.9 *Let p be a propositional variable and $\mu, \lambda, \theta \in |\tau|$. Then, the following hold:*

1. $\vdash p_{\lambda\vee\mu} \to p_\lambda$
2. $\vdash p_{\lambda\vee\mu} \to p_\mu$
3. $\lambda \geq \mu$ *and* $\lambda \geq \theta \Rightarrow \vdash p_\lambda \to p_{\mu\vee\theta}$
4. $\vdash p_\mu \to p_{\mu\wedge\theta}$.
5. $\vdash p_\theta \to p_{\mu\wedge\theta}$.
6. $\lambda \leq \mu$ *and* $\lambda \leq \theta \Rightarrow \vdash p_{\mu\wedge\theta}$
7. $\vdash p_\mu \leftrightarrow p_{\mu\vee\mu}$, $\vdash p_\mu \leftrightarrow p_{\mu\wedge\mu}$
8. $\vdash p_{\mu\vee\lambda} \leftrightarrow p_{\lambda\vee\mu}$, $\vdash p_{\mu\wedge\lambda} \leftrightarrow p_{\lambda\wedge\mu}$
9. $\vdash p_{(\mu\vee\lambda)\vee\theta}\vee \to p_{\mu\vee(\lambda\vee\theta)}$, $\vdash p_{(\mu\wedge\lambda)\wedge\theta}\vee \to p_{\mu\wedge(\lambda\wedge\theta)}$
10. $p_{(\mu\vee\lambda)\wedge\mu} \to p_\mu$, $p_{(\mu\wedge\lambda)\vee\mu} \to p_\mu$
11. $\lambda \leq \mu \Rightarrow \vdash p_{\lambda\vee\mu} \to p_\mu$
12. $\lambda \vee \mu = \mu \Rightarrow \vdash p_\mu \to p_\lambda$
13. $\mu \geq \lambda \Rightarrow \forall\theta \in |\tau|$ $(\vdash p_{\mu\vee\theta} \to p_{\lambda\vee\theta}$ *and* $\vdash p_{\mu\wedge\theta} \to p_{\lambda\wedge\theta})$
14. $\mu \geq \lambda$ *and* $\theta \geq \varphi \Rightarrow \vdash p_{\mu\vee\theta} \to p_{\lambda\vee\varphi}$ *and* $p_{\mu\wedge\theta} \to p_{\lambda\wedge\varphi}$
15. $\vdash p_{\mu\wedge(\lambda\vee\theta)} \to p_{(\mu\wedge\lambda)\vee(\mu\wedge\theta)}$, $\vdash p_{\mu\vee(\lambda\wedge\theta)} \to p_{(\mu\vee\lambda)\wedge(\mu\vee\theta)}$
16. $\vdash p_\mu \wedge p_\lambda \leftrightarrow p_{\mu\wedge\lambda}$
17. $\vdash p_{\mu\vee\lambda} \to p_\mu \vee p_\lambda$

Example 5.1 Consider the complete lattice $\tau = N \cup \{\omega\}$, where N is the set of natural numbers. The ordering on τ is the usual ordering on ordinals, restricted to the set τ. Consider the set $\Gamma = \{p_0, p_1, p_2, \ldots\}$, where $p_\omega \notin \Gamma$. It is clear that $\Gamma \vdash p_\omega$, but an infinitary deduction is required to establish this.

Definition 5.12 $\overline{\Delta} = \{A \in \mathbf{F} \mid \Delta \vdash A\}$

Definition 5.13 Δ is said to be *trivial* iff $\overline{\Delta} = \mathbf{F}$ (i.e., every formula in our language is a syntactic consequence of Δ); otherwise, Δ is said to be *non-trivial*. Δ is said to be *inconsistent* iff there is some formula A such that $\Delta \vdash A$ and $\Delta \vdash \neg A$; otherwise, Δ is *consistent*.

From the definition of triviality, the next theorem follows:

Theorem 5.11 *Δ is trivial iff $\Delta \vdash A \wedge \neg A$ (or $\Delta \vdash A$ and $\Delta \vdash \neg_* A$) for some formula A.*

Theorem 5.12 *Let Γ be a set of formulas, A, B be any formulas, and F be any complex formula. Then, the following hold.*

1. $\Gamma \vdash A$ *and* $\Gamma \vdash A \rightarrow B \;\Rightarrow\; \Gamma \vdash B$
2. $A \wedge B \vdash A$
3. $A \wedge B \vdash B$
4. $A, B \vdash A \wedge B$
5. $A \vdash A \vee B$
6. $B \vdash A \vee B$
7. $\Gamma, A \vdash C$ *and* $\Gamma, B \vdash C \;\Rightarrow\; \Gamma, A \vee B \vdash C$
8. $\vdash F \leftrightarrow \neg_* F$
9. $\Gamma, A \vdash B$ *and* $\Gamma, A \vdash \neg_* B \;\Rightarrow\; \Gamma \vdash \neg_* A$
10. $\Gamma, A \vdash B$ *and* $\Gamma, \neg_* A \vdash B \Rightarrow \; \Gamma \vdash B$.

Note here that the counterpart of Theorem 5.12 (10) obtained by replacing the occurrence of $\neg_*$ by $\neg$ is not valid.

Now, we are in a position to prove the soundness and completeness of $P\tau$. Our proof method for completeness is based on maximal non-trivial set of formulas; see Abe [1] and Abe and Akama [6]. da Costa et al. [30] presented another proof using Zorn's Lemma.

Theorem 5.13 (Soundness) *Let Γ be a set of formulas and A be any formula. $\mathcal{A}\tau$ is a sound axiomatization of $P\tau$, i.e., if $\Gamma \vdash A$ then $\Gamma \models A$.*

For proving the completeness theorem, we need some theorems.

Theorem 5.14 *Let Γ be a non-trivial set of formulas. Suppose that τ is finite. Then, Γ can be extended to a maximal (with respect to inclusion of sets) non-trivial set with respect to* **F**.

Theorem 5.15 *Let Γ be a maximal non-trivial set of formulas. Then, we have the following:*

1. *if A is an axiom of $P\tau$, then $A \in \Gamma$*
2. *$A, B \in \Gamma$ iff $A \wedge B \in \Gamma$*
3. *$A \vee B \in \Gamma$ iff $A \in \Gamma$ or $B \in \Gamma$*
4. *if $p_\lambda, p_\mu \in \Gamma$, then $p_\theta \in \Gamma$, where $\theta = max(\lambda, \mu)$*
5. *$\neg^k p_\mu \in \Gamma$ iff $\neg^{k-1} p_{\sim\mu} \in \Gamma$, where $k \geq 1$*
6. *if $A, A \rightarrow B \in \Gamma$, then $B \in \Gamma$*
7. *$A \rightarrow B \in \Gamma$ iff $A \notin \Gamma$ or $B \in \Gamma$*

Theorem 5.16 *Let Γ be a maximal non-trivial set of formulas. Then, the characteristic function χ of Γ, that is, $\chi_\Gamma \rightarrow \mathbf{2}$ is the valuation function of some interpretation $I : \mathbf{P} \rightarrow |\tau|$.*

Here is the completeness theorem for $P\tau$.

Theorem 5.17 (Completeness) *Let Γ be a set of formulas and A be any formula. If τ is finite, then $\mathcal{A}\tau$ is a complete axiomatization for $P\tau$, i.e., if $\Gamma \models A$ then $\Gamma \vdash A$.*

The decidability theorem also holds for finite lattice.

Theorem 5.18 (Decidability) *If τ is finite, then $P\tau$ is decidable.*

The completeness does not in general hold for infinite lattice. But, it holds for special case.

Definition 5.14 (*Finite annotation property*) Suppose that Γ be a set of formulas such that the set of annotated constants occurring in Γ is included in a finite substructure of τ (Γ itself may be infinite). In this case, Γ is said to have the *finite annotation property*.

Note that if τ' is a substructure of τ then τ' is closed under the operations $\sim$, $\vee$ and $\wedge$. One can easily prove the following from Theorem 5.17.

Theorem 5.19 (Finitary Completeness) *Suppose that Γ has the finite annotation property. If A is any formula such that $\Gamma \vdash A$, then there is a finite proof of A from Γ.*

Theorem 5.19 tells us that even if the set of the underlying truth-values of $P\tau$ is infinite (countably or uncountably), as long as theories have the finite annotation property. The completeness result applied to them, i.e., $\mathcal{A}\tau$ is complete with respect to such theories.

In general, when we consider theories that do not possess the finite annotation property, it may be necessary to guarantee completeness by adding a new infinitary inference rule (ω-rule), similar in spirit to the rule used by da Costa [24] in order to cope with certain models in a particular family of infinitary language. Observe that for such cases a desired axiomatization of $P\tau$ is not finitary.

From the classical result of compactness, we can state a version of the compactness theorem.

Theorem 5.20 (Weak Compactness) *Suppose that Γ has the finite annotation property. If A is any formula such that $\Gamma \vdash A$, then there is a finite subset Γ' of Γ such that $\Gamma' \vdash A$.*

Annotated logics $P\tau$ provide a general framework, and can be used to reasoning about many different logics. Below we present some examples.

The set of truth-values $FOUR = \{t, f, \bot, \top\}$, with $\neg$ defined as: $\neg t = f, \neg f = t, \neg\bot = \bot, \neg\top = \top$. Four-valued logic based on $FOUR$ was originally due to Belnap [20, 21] to model internal states in a computer.

Subrahmanian [45] formalized an annotated logic with $FOUR$ as a foundation for paraconsistent logic programming; also see Blair and Subrahmanian [22].

Their annotated logic may be used for reasoning about inconsistent knowledge bases. For example, we may allow logic programs to be finite collections of formulas of the form:

$$(A : \mu_0) \leftrightarrow (B_1 : \mu_1) \& \cdots \& (B_n : \mu_n)$$

where A and B_i $(1 \leq i \leq n)$ are atoms and μ_j $(0 \leq j \leq n)$ are truth-values in $FOUR$.

Intuitively, such programs may contain "intuitive" inconsistencies–for example, the pair

$$((p : f), (p : t))$$

is inconsistent. If we append this program to a consistent program P, then the resulting union of these two programs may be inconsistent, even though the predicate symbols p occurs nowhere in program P.

Such inconsistencies can easily occur in knowledge based systems, and should not be allowed to trivialize the meaning of a program. However, knowledge based systems based on classical logic cannot handle the situation since the program is trivial.

In Blair and Subrahmanian [22], it is shown how the four-valued annotated logic may be used to describe this situation. Later, Blair and Subrahmanian's annotated logic was extended as *generalized annotated logics* by Kifer and Subrahmanian [37].

There are also other examples which can be dealt with by annotated logics. The set of truth-values $FOUR$ with negation defined as boolean complementation forms an annotated logic.

The unit interval [0, 1] of truth-values with $\neg x = 1 - x$ is considered as the base of annotated logic for qualitative or fuzzy reasoning. In this sense, probabilistic and fuzzy logics could be generalized as annotated logics.

The interval $[0, 1] \times [0, 1]$ of truth-values can be also used for annotated logics for evidential reasoning. Here, the assignment of the truth-value (μ_1, μ_2) to proposition p may be thought of as saying that the degree of belief in p is μ_1, while the degree of disbelief is μ_2. Negation can be defined as $\neg(\mu_1, \mu_2) = (\mu_2, \mu_1)$.

Note that the assignment of $[\mu_1, \mu_2]$ to a proposition p by an interpretation I does not necessarily satisfy the condition $\mu_1 + \mu_2 \leq 1$. This contrasts with probabilistic reasoning. Knowledge about a particular domain may be gathered from different experts (in that domain), and these experts may different views.

Some of these views may lead to a "strong" belief in a proposition; likewise, other experts may have a "strong" disbelief in the same proposition. In such a situation, it seems appropriate to report the existence of conflicting opinions, rather than use ad-hoc means to resolve this conflict.

5.3 Predicate Annotated Logics $Q\tau$

As mentioned above, da Costa et al. [30] investigated propositional annotated logics $P\tau$, and suggested their predicate extension $Q\tau$ (also denoted $Q\mathcal{T}$). We can look at the detailed formulation of $Q\tau$ in da Costa et al. [26]; also see Abe [1].

Predicate annotated logics $Q\tau$ can be formalized as a two-sorted first-order logic. We repeat some definitions below. $\tau = \langle |\tau|, \leq, \sim \rangle$ is some arbitrary, but fixed complete lattice, with the ordering $\leq$ and the operator $\sim : |\tau| \to |\tau|$. The bottom element of this lattice is denoted by $\bot$, and top element is denoted by $\top$.

The language L^τ of $Q\tau$ is a first-order language without equality. Abe [1] introduced equality into $Q\tau$.

Definition 5.15 (*Symbols*) Primitive symbols are the following:

1. Logical connectives: $\wedge$ (conjunction), $\vee$ (disjunction), $\rightarrow$ (implication), and $\neg$ (negation)
2. Individual variables: a denumerably infinite set of variable symbols
3. Individual constants: an arbitrary family of constant symbols
4. Quantifiers: $\forall$ (for all) and $\exists$ (exists)
5. Function symbols: for each natural number $n > 0$, a collection of function symbols of arity n
6. Annotated predicate symbols: for any natural number $n \geq 0$, and any $\lambda \in \tau$, a family of annotated predicate symbols p_τ
7. Parentheses: (and)

Here, $\forall$ is called the *universal quantifier* and $\exists$ the *existential quantifier*. We define the notion of *term* as usual. Given an annotated predicate symbol p_λ of arity n and n terms $t_1, \ldots, t_n$, an *annotated atom* is an expression of the form $p_\lambda(t_1, \ldots, t_n)$.

Definition 5.16 (*Formulas*) Formulas are defined as follows:

1. An annotated atom is a formula.
2. If F is a formula, then $\neg F$ is a formula.
3. If F and G are formulas, then $F \wedge G$, $F \vee G$, $F \rightarrow G$ are formulas.
4. If F is a formula and x is an individual variable, then $\forall x F$ and $\exists x F$ are formulas.

Definition 5.17 (*Hyper-literal and complex formulas*) Hyper-literal and complex formulas are defined as follows. A formula of the form $\neg^k p_\mu(t_1, \ldots, t_n)$ $(k \geq 0)$ is called a *hyper-literal*. A formula which is not a hyper-literal is called a *complex formula*

As in $P\tau$, we may also use the formulas of the form $A \leftrightarrow B$ and $\neg_* A$ in $Q\tau$. Here, $\leftrightarrow$ denotes the *equivalence* and $\neg_*$ *strong negation*, respectively. We can also introduce the *equality*, denoted $=$, into $Q\tau$. If t and s are terms, then $s = t$ is also a formula. $s = t$ is read "s and t are equal".

Now, we describe a semantics for $Q\tau$, which is a variant of the semantics for standard first-order logic.

Definition 5.18 (*Interpretation*) An *interpretation* I for the language L^τ of $Q\tau$ consists of a non-empty set, denoted by $dom(I)$, and called the *domain*, together with

1. a function η_I that maps constants of L^τ to $dom(I)$
2. a function ζ_I that assigns, to each function symbol f of arity n in L^τ, a function from $(dom(I))^n$ to $dom(I)$
3. a function χ_I that assigns, to each predicate symbol of arity n in L^τ, a function from $(dom(I))^n$ to τ.

Definition 5.19 (*Variable assignment*) Suppose I is an interpretation for L^τ. Then, a *variable assignment* v for L^τ with respect to I is a map from the set of variables symbols of L^τ to $dom(I)$.

Definition 5.20 (*Denotation*) The *denotation* $d_{I,v}(t)$ of a term t with reference to an interpretation I and variable assignment v is defined inductively as follows:

1. If t is a constant symbol, then $d_{I,v}(t) = \eta(t)$.
2. If t is a variable symbol, then $d_{I,v}(t) = v(t)$.
3. If t is a function symbol, then $d_{I,v}(t) = \zeta(f)(d_{I,v}(t_1), \ldots, d_{I,v}(t_n))$.

Definition 5.21 (*Truth relation*) Let I and v be an interpretation of L^τ and a variable assignment with reference to I, respectively. We also suppose that A is an ordinary atom, and that F, G and H are any formulas whatsoever. Then, the *truth relation* $I, v \models A$, saying that A is true with reference to an interpretation I and variable assignment v, is defined as follows:

1. $I, v \models p_\mu(t_1, \ldots, t_n)$ iff $\chi_I(p)(d_{I,v}(t_1), \ldots, d_{I,v}(t_n)) \geq \mu$
2. $I, v \models \neg^k A_\mu$ iff $I, v \models \neg^{k-1} A_{\sim\mu}$
3. $I, v \models F \wedge G$ iff $I, v \models F$ and $I, v \models G$
4. $I, v \models F \vee G$ iff $I, v \models F$ or $I, v \models G$
5. $I, v \models F \rightarrow G$ iff $I, v \not\models F$ or $I, v \models G$
6. $I, v \models \neg F$ iff $I, v \not\models F$, where F is not a hyper-literal
7. $I, v \models \exists x H$ iff for some variable assignment v' such that for all variables y different from x, $v(y) = v'(y)$, we have that $I, v' \models H$
8. $I, v \models \forall x H$ iff for all variable assignments v' such that for all variables y different from x, $v(y) = v'(y)$, we have that $I, v' \models H$
9. $I \models H$ iff for all variable assignments v associated with I, $I, v \models H$

The equality $s = t$ is interpreted as follows:

$I, v \models s = t$ iff $d_{I,v}(s) = d_{I,v}(t)$

Here, $=$ at the right hand side of 'iff' denotes the equality symbol in the meta-language, and it reads classically. We could also introduce annotated quality $=_\lambda$ as a binary annotated atom. However, we do not go into details here.

We can define the notions of validity, model and semantic consequence as in Sect. 5.2. Let $\Gamma \cup \{H\}$ be a set of formulas. We write $\models H$, and say that H is *valid* (in $Q\tau$) if, for every interpretation I, $I \models H$. If $I \models A$ for each $A \in \Gamma$, I is a *model* of Γ. We say that H is a *semantic consequence* of Γ iff for any interpretation I such that $I \models G$ for all $G \in \Gamma$, it is the case that $I \models F$.

The following lemmas concerns the properties of $\models$, whose proofs are immediate from the corresponding proof in the previous chapter.

Lemma 5.10 *For any complex formula A and B and any formula F, the valuation v satisfies the following:*

1. $\models A \leftrightarrow B$ iff $\models A \rightarrow B$ and $\models B \rightarrow A$
2. $\not\models (A \rightarrow A) \wedge \neg(A \rightarrow A)$
3. $\models \neg_* A$ iff $\not\models A$
4. $\models \neg F \leftrightarrow \neg_* F$

Lemma 5.11 *Let $p_\mu(t_1, \ldots, t_n)$ be an annotated atom and $\mu, \lambda \in |\tau|$. Then, we have:*

1. $\models p_\perp(t_1, \ldots, t_n)$
2. $\models p_\mu(t1, \ldots, t_n) \rightarrow p_\lambda(t_1, \ldots, t_n),\ \mu \geq \lambda$
3. $\models \neg^k p_\mu(t_1, \ldots, t_n) \leftrightarrow \neg^{k-1} p_{\sim\mu}(t_1, \ldots, t_n),\ k \geq 0$

Next, we show a Hilbert style axiomatization of $Q\tau$, called $\mathcal{A}$. In the formulation of the postulates of $\mathcal{A}$, the symbols A, B, C denote any formula whatsoever, F and G denote complex formulas, and P_λ is an annotated atom.

Postulates for $\mathcal{A}$ described in Abe [1] are as follows; also see da Costa et al. [26].

Postulates for $\mathcal{A}$

($\rightarrow_1$) $(A \rightarrow (B \rightarrow A)$
($\rightarrow_2$) $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$
($\rightarrow_3$) $((A \rightarrow B) \rightarrow A) \rightarrow A$
($\rightarrow_4$) $A, A \rightarrow B/B$
($\wedge_1$) $(A \wedge B) \rightarrow A$
($\wedge_2$) $(A \wedge B) \rightarrow B$
($\wedge_3$) $A \rightarrow (B \rightarrow (A \wedge B))$
($\vee_1$) $A \rightarrow (A \vee B)$
($\vee_2$) $B \rightarrow (A \vee B)$
($\vee_3$) $(A \rightarrow C) \rightarrow ((B \rightarrow C) \rightarrow ((A \vee B) \rightarrow C))$
($\neg_1$) $(F \rightarrow G) \rightarrow ((F \rightarrow \neg G) \rightarrow \neg F)$
($\neg_2$) $F \rightarrow (\neg F \rightarrow A)$
($\neg_3$) $F \vee \neg F$
($\exists_1$) $A(t) \rightarrow \exists x A(x)$
($\exists_2$) $A(x) \rightarrow B/\exists x A(x) \rightarrow B$
($\forall_1$) $\forall x A(x) \rightarrow A(t)$
($\forall_2$) $A \rightarrow B(x)/A \rightarrow \forall x B(x)$
(τ_1) $p_\perp(a_1, \ldots, a_n)$
(τ_2) $\neg^k p_\lambda(a_1, \ldots, a_n) \leftrightarrow \neg^{k-1} p_{\sim\lambda}(a_1, \ldots, a_n)$
(τ_3) $p_\lambda(a_1, \ldots, a_n) \rightarrow p_\mu(a_1, \ldots, a_n)$, where $\lambda \geq \mu$
(τ_4) If $A \rightarrow p_{\lambda_j}(a_1, \ldots, a_n)$, then $A \rightarrow p_\lambda(a_1, \ldots, a_n)$ for every $j \in J$, where$\lambda = \bigvee_{i=1}^{m} \lambda_i$

As τ is a complete lattice, the supremum in (τ_4) is well-defined. The postulates for quantifiers are subject to the usual restrictions. When τ is finite, (τ_4) can be replaced by the schema:

$$p_{\lambda_1}(a_1, \ldots, a_n) \wedge p_{\lambda_2}(a_1, \ldots, a_n) \wedge \cdots \wedge p_{\lambda_m}(a_1, \ldots, a_n) \rightarrow p_\lambda(a_1, \ldots, a_n),$$
where $\lambda = \bigvee_{i=1}^{m} \lambda_i$

Here, ($\rightarrow_4$), ($\exists_4$), ($\forall_4$) and (τ_4) are regarded as rules of inference

Abe [1] also added the following three axioms for equality:

($=_1$) $x = x$
($=_2$) $x_1 = y_1 \rightarrow (\ldots \rightarrow (x_n = y_n \rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)))$
($=_3$) $x_1 = y_1 \rightarrow (\ldots \rightarrow (x_n = y_n \rightarrow P(x_1, \ldots, x_n) \rightarrow P(y_1, \ldots, y_n)))$

Here, f and P are function symbol and predicate symbol, respectively.

As in $\mathcal{A}\tau$, we easily define the syntactic concepts related to $\mathcal{A}$; in particular the concepts of syntactic consequence $\vdash$ is defined in the normal way. We only note that the notion of deduction (proof) is not finitary if τ is infinite.

da Costa, Abe and Subrahmanian's axiomatization of $\mathcal{A}$ adopts different naming for postulates, but it is equivalent to the above axiomatization. The deduction theorem (Theorem 2.8) also holds for $\mathcal{A}$.

Theorem 5.21 *The following dualities of quantifiers hold:*

1. $\vdash \forall x A \leftrightarrow \neg_* \exists x \neg_* A$
2. $\vdash \exists x A \leftrightarrow \neg_* \forall x \neg_* A$

Here are some formal results of $Q\tau$. The first result is soundness of $Q\tau$.

Theorem 5.22 (Soundness) *Let $\Gamma \cup \{A\}$ be a set of formulas of $Q\tau$. Then, $\Gamma \vdash A$ (in $\mathcal{A}$) implies that $\Gamma \vdash A$, i.e., $\mathcal{A}$ is sound with respect to the semantics of $Q\tau$.*

The next result is completeness of $Q\tau$ in a restricted sense.

Theorem 5.23 (Completeness) *Let $\Gamma \cup \{A\}$ be a set of formulas of $Q\tau$. Then, if τ is finite or if $\Gamma \cup \{A\}$ possesses the finite annotation property, we have that $\Gamma \models A$ entails $\Gamma \vdash A$, i.e., $\mathcal{A}$ is complete with respect to the semantics of $Q\tau$.*

When τ is infinite, it seems that completeness can be obtained only by augmenting $\mathcal{A}$ with an extra infinitary rule.

$Q\tau$ belong to the class of non-classical logics, and they are paraconsistent and paracomplete. They have a weak negation $\neg$, but we can define the strong negation $\neg_*$, which is classical.

da Costa et al. [26] presented another axiomatization of $Q\tau$ with a different nature, which is obtained by adjoining to the classical first-order logic, a weak negation $\neg$ plus some extra convenient postulates.

Let $\mathcal{C}$ be an axiomatization of classical first-order logic (without equality), in which negation is denoted by $\sim$. The remaining primitives defined symbols of $\mathcal{C}$ are the same as the corresponding one of $Q\tau$. We also suppose that the atomic formulas of the language of $\mathcal{C}$ are annotated atoms, as in L^τ. Furthermore, we suppose that we have added to $\mathcal{C}$ a weak negation $\neg$.

We denote by $\mathcal{A}'$ the axiomatic system obtained from $\mathcal{C}$ by adding the axioms $(\neg_1)$, $(\neg_2)$, $(\neg_3)$, (τ_1), (τ_2), (τ_3), (τ_4), and the rule:

> If F and G are formulas such that G is obtained from F by the replacement of a sub-formula of the form $\neg_* A$ by $A \to (A \to A) \wedge \neg(A \to A)$ or by the replacement of a sub-formula of the latter from by one of the first form, then infer $F \leftrightarrow G$.

Theorem 5.24 *$\mathcal{A}$ and $\mathcal{A}'$ are equivalent; both characterize $Q\tau$.*

Theorem 5.24 reveals that annotated logic $Q\tau$ can be interpreted as an extension of classical first-order logic C. This fact seems interesting theoretically as well as practically.

Annotated logics can be used for various mathematical subjects. For example, it is possible to work out a set theory based on $Q\tau$. We will explore annotated set theory. For this purpose, we need the notion of normal structure, and need to define a fragment of $Q\tau$.

Definition 5.22 (*Normal structure*) Let X be a non-empty set. A *normal structure* based on X is a function $f : X \times X \to \tau$.

We denote by $Q\tau^2$ the logic $Q\tau$ obtained by suppressing all function symbols and all predicate symbols, with the exception of one predicate symbol of arity 2 (a binary predicate symbol) which we represent by $\in$. $A\tau^2$ is then a dyadic predicate calculus whose atoms are annotated by τ. An annotated atom of $Q\tau^2$ has the form $\in_\lambda (a, b)$, where a and b are terms and $\lambda \in \tau$. This atom will be written $a \in_\lambda b$.

Intuitively, $\in$ is the membership predicate symbol. The subscript λ denotes a "degree" of membership. A normal structure is basically just a first-order interpretation as defined earlier with the following differences. First, $Q\tau^2$ contains only one predicate symbol $\in$ associated with different members of τ. Second, the normal structures are the interpretations of $\in$.

Theorem 5.25 *$Q\tau^2$ is sound with respect to the semantics of normal structures. If τ is finite or we consider only sets of formulas sharing the finite annotation property, then $Q\tau^2$ is also complete.*

5.4 Curry Algebras

We can develop an algebraic semantics for $P\tau$. Algebraic semantics is mathematically more elegant than model-theoretic semantics. However, algebraic semantics for paraconsistent logics challenges standard formulation, since known techniques cannot be properly used. Abe [3] proposed Curry algebra $P\tau$ that algebraizes propositional annotated logics $P\tau$. Abe proved the completeness theorem for $P\tau$ with respect to the algebraic semantics.

In order to obtain algebraic versions of the majority of logical systems the procedure is the following: we define an appropriate equivalence relation in the set of formulas (e.g. identifying equivalent formulas in classical propositional logic), in such a way that the primitive connectives are compatible with the equivalence relation, i.e., a congruence.

The resulting quotient system is the algebraic structure linked with the corresponding logical system. By this process, Boolean algebra constitutes the algebraic version of classical propositional logic, Heyting algebra constitutes the algebraic version of intuitionistic propositional logic, and so on. Thus, the procedure is to formulate an algebraic semantics.

However, in some non-classical logics, it is not always clear what "appropriate" equivalence relation here can be; the non-existence of any significant equivalence relation among formulas of the calculus can also take place. This occurs, for instance, with some paraconsistent systems; see Mortensen [39]. Indeed, as pointed out by Eytan [32], even for classical logic, it may not always be convenient to apply these ideas.

Now, we give some basic definitions related to Curry algebras. In $P\tau$, we define $A \leq B$ by setting $\vdash A \rightarrow B$, and $A \equiv B$ by setting $A \leq B$ and $B \leq A$. Here, $\leq$ is a quasi-order and $\equiv$ is an equivalence relation, respectively. Let R be a set whose elements are denoted by x, y, z, x', y'.

Definition 5.23 (*Curry pre-ordered system*) A system $(R, \equiv, \leq)$ is called a *Curry pre-ordered system*, if

1. $\equiv$ is an equivalence relation on R
2. $x \leq x$
3. $x \leq y$ and $y \leq z$ imply $x \leq z$
4. $x \leq y, x' \equiv$ and $y' \equiv y$ imply $x' \leq y'$.

Definition 5.24 (*Pre-lattice*) A system $(R, \equiv, \leq)$ is called a *pre-lattice*, if $(R, \equiv, \leq)$ is a Curry pre-ordered system and

1. $\inf\{x, y\} \neq \emptyset$
2. $\sup\{x, y\} \neq \emptyset$.

We denote by $x \wedge y$ one element of the set of $\inf\{x, y\}$ and by $x \vee y$ one element of the set of $\sup\{x, y\}$.

Definition 5.25 (*Implicative pre-lattice*) A system $(R, \equiv, \leq)$ is called a *implicative pre-lattice*, if

1. $(R, \equiv, \leq)$ is a pre-lattice
2. $x \wedge (x \rightarrow y) \leq y$
3. $x \wedge y \leq z$ iff $x \leq y \rightarrow z$.

Definition 5.26 An *implicative pre-lattice* $(R, \equiv, \leq)$ is called *classic* if $(x \rightarrow y) \rightarrow x \leq y$ (Peirce's law).

As is obvious from the above definitions, a classic implicative pre-lattice is a pre-algebraic structure which can characterize positive classical propositional logic, i.e., classical propositional logic without negation. As is well known, Peirce's law corresponds to the law of excluded middle.

We are now ready to define a Curry algebra $P\tau$. Let S be a non-empty set and $\tau = (|\tau|, \leq)$ be a finite lattice with the operation $\sim : |\tau| \rightarrow |\tau|$. We denote by S^* the set of all pairs (p, λ), where $p \in S$ and $\lambda \in |\tau|$.

We now consider the set $S^* \cup \{\neg, \wedge, \vee, \rightarrow\}$. Let S^{**} be the smallest algebraic structure freely generated by the set $S^* \cup \{\neg, \wedge, \vee, \rightarrow\}$ by the usual algebraic method. Elements of S^{**} are classified in two categories: *hyper-literal elements* are of the form $\neg^k(p, \lambda)$ and *complex elements* are the remaining elements of S^{**}.

Now, we introduce the concept of a Curry algebra $P\tau$.

Definition 5.27 (*Curry algebra $P\tau$*) A *Curry algebra $P\tau$* (abbreviated by $P\tau$-algebra) is a structure $R\tau = (R, (|\tau|, \leq, \sim), \equiv, \rightarrow, \neg)$ and, for $p \in R$, $a \in R^*$, $x, y \in R^{**}$:

1. R^{**} is a classical implicative lattice with a greatest element 1
2. $\neg$ is a unary operator $\neg : R^{**} \to R^{**}$
3. $x \to y \leq (x \to \neg y) \to \neg x$
4. $x \leq \neg x \to a$
5. $p_\perp \equiv 1$
6. $x \vee \neg x \equiv 1$
7. $\neg^k (p, \lambda) \equiv \neg^{k-1}(p, \sim \lambda), k \geq 1$
8. If $\mu \leq \lambda$ then $(p, \mu) \leq (p, \lambda)$
9. $(p, \lambda_1) \wedge (p, \lambda_2) \wedge \cdots \wedge (p, \lambda_n) \leq (p, \lambda)$, where $\lambda = \bigvee_{i=1}^{n} \lambda_i$

One can easily see that a $P\tau$-algebra is distributive and has a greatest element as well as a first element.

Definition 5.28 Let x be an element of a $P\tau$-algebra. We put:

$$\neg_* x = x \to ((x \to x) \wedge \neg(x \to x))$$

In a $P\tau$-algebra, $\neg_* x$ is a Boolean complement of x, so both $x \vee \neg_* x \equiv 1$ and $x \wedge \neg_* x \equiv 0$ hold.

Theorem 5.26 *In a $P\tau$-algebra, the structure composed by the underlying set and by operations $\wedge, \vee$ and $\neg_*$ is a pre-Boolean algebra. If we pass to the quotient through the basic relation $\equiv$, we obtain a Boolean algebra in the usual sense.*

A *pre-Boolean algebra* is a partial preorder $(R, \leq)$ such that the quotient by the relation $\equiv$. Thus, by definition of $P\tau$-algebra, the mentioned structure is a pre-Boolean algebra.

In addition, replacing the class of equivalent formulas by a formula can produce a usual Boolean algebra in which the meet $\wedge$ is conjunction, the join $\vee$ is disjunction, and the complement is negation.

Definition 5.29 Let $(R, (|\tau|, \leq, \sim), \equiv, \leq, \to, \neg)$ be a $P\tau$-algebra, and $(R, (|\tau|, \leq, \sim), \equiv, \leq, \to, \neg_*)$ the Boolean algebra that is isomorphic to the quotient algebra of $(R, (|\tau|, \leq, \sim), \equiv, \leq, \to, \neg_*)$ by $\equiv$ is called the Boolean algebra *associated with* the $P\tau$-algebra.

Hence, we can establish the following first representation theorem for $P\tau$-algebra.

Theorem 5.27 *Any $P\tau$-algebra is associated with a field of sets. Moreover, any $P\tau$-algebra is associated with the field of sets simultaneously open and closed of a totally disconnected compact Hausdorff space.*

This is not the only way of extracting Boolean algebra out of $P\tau$-algebra. There is another natural Boolean algebra associated with a $P\tau$-algebra.

Definition 5.30 Let $(R, (|\tau|, \leq, \sim), \equiv, \leq, \rightarrow, \neg)$ be a $P\tau$-algebra. By RC we indicate the set of all complex elements of $(R, (|\tau|, \leq, \sim), \equiv, \leq, \rightarrow, \neg)$.

Then, the structure $(RC, (|\tau|, \leq, \sim), \equiv, \leq, \rightarrow, \neg)$ constitutes a pre-Boolean algebra which we call Boolean algebra *c-associated with* the $P\tau$-algebra $(R, (|\tau|, \leq, \sim), \equiv, \leq, \rightarrow, \neg)$. Thus, we obtain a second representation theorem for $P\tau$-algebra.

Theorem 5.28 *Any $P\tau$-algebra is c-associated with a field of sets. Moreover, any $P\tau$-algebra is c-associated with the field of sets simultaneously open and closed of a totally disconnected compact Hausdorff space.*

Theorems 5.27 and 5.28 show us that $P\tau$-algebra constitute interesting generalizations of the concept of Boolean algebra. There are some open questions related to these results. How many non-isomorphic Boolean algebra associated with a $P\tau$-algebra is there? How many non-isomorphic Boolean algebra c-associated with a $P\tau$-algebra is there? The answers to these questions can establish connections of associated and c-associated algebra.

Next, we show soundness and completeness of $P\tau$-algebras using the notion of filter and ideal of a $P\tau$-algebra.

Definition 5.31 (*Filter*) Let $(R, (|\tau|, \leq, \sim), \equiv, \leq, \rightarrow, \neg)$ be a $P\tau$-algebra. A subset F of R is called a *filter* if:

1. $x, y \in F$ imply $x \wedge y \in F$
2. $x \in F$ and $y \in R$ imply $x \vee y \in F$
3. $x \in F$, $y \in R$, and $x \equiv y$ imply $y \in F$.

Definition 5.32 (*Ideal*) Let $(R, (|\tau|, \leq, \sim), \equiv, \leq, \rightarrow, \neg)$ be a $P\tau$-algebra. A subset I of R is called an *ideal* if:

1. $x, y \in I$ imply $x \vee y \in I$
2. $x \in I$ and $y \in R$ imply $x \wedge y \in F$
3. $x \in I$, $y \in R$, and $x \equiv y$ imply $y \in F$.

Then, we have the following lemma whose proof is trivial.

Lemma 5.12 *Let $(R, (|\tau|, \leq, \sim), \equiv, \leq, \rightarrow, \neg)$ be a $P\tau$-algebra. A subset F of R is a filter iff:*

1. *$x, y \in F$ imply $x \wedge y \in F$*
2. *$x \in F$, $y \in R$, and $x \leq y$ imply $y \in F$*
3. *$x \in F$, $y \in R$, and $x \equiv y$ imply $y \in F$.*

A subset I of R is an ideal iff:

1. $x, y \in I$ *imply* $x \vee y \in I$
2. $x \in I, y \in R$, *and* $x \leq y$ *imply* $y \in I$
3. $x \in I, y \in R$, *and* $x \equiv y$ *imply* $y \in I$.

Filters are partially ordered by inclusion. Filters that are maximal with respect to this ordering are called *ultrafilters*. By the Ultrafilter Theorem, every filter in $P\tau$-algebra can be extended to an ultrafilter.

Theorem 5.29 *Let F be an ultrafilter in a $P\tau$-algebra. Then, we have:*

1. $x \wedge y \in F$ *iff* $x \in F$ *and* $y \in F$
2. $x \vee y \in F$ *iff* $x \in F$ *or* $y \in F$
3. $x \to y \in F$ *iff* $x \notin F$ *or* $y \in F$
4. *If* $p_{\lambda_1}, p_{\lambda_2} \in F$, *then* $p_\lambda \in F$, *where* $\lambda = \lambda_1 \vee \lambda_2$
5. $\neg^k p_\lambda \in F$ *iff* $\neg^{k-1} p_{\sim\lambda} \in F$
6. *If* $x, x \to y \in F$, *then* $y \in F$

Definition 5.33 If $R\tau_1 = (R_1, (|\tau_1|, \leq_1, \sim_1), \equiv_1, \leq_1, \to_1, \neg_1)$ and $R\tau_2 = (R_2, (|\tau_2|, \leq_2, \sim_2), \equiv_2, \leq_2, \to_2, \neg_2)$ are two $P\tau$-algebras, then a homomorphism of $R\tau_1$ into $R\tau_2$ is a map f of R_1 into R_2 which preserves the algebraic operations, i.e., such that for $x, y \in R_1$:

1. $x \leq_1 y$ iff $f(x) \leq_2 f(y)$
2. $f(x \to_1 y) \equiv_2 f(x) \to_2 f(y)$
3. $f(\neg_1 x) \equiv_2 \neg_2 f(x)$
4. If $x \equiv_1 y$, then $f(x) \equiv_2 f(y)$
5. f is also extended to a homomorphism of $(|\tau_1|, \leq_1, \sim_1)$ into $(|\tau_2|, \leq_2, \sim_2)$ in an obvious way (i.e., for instance, $f(\sim_1 \lambda) = \sim_2 f(\lambda)$).

Then, as in the classical case, we can present the following theorem:

Theorem 5.30 *Let $R\tau_1$ and $R\tau_2$ be two $P\tau$-algebras and f a homomorphism from $R\tau_1$ into $R\tau_2$. Then, the set $\{x \in R_1 \mid f(x) \equiv_2 1_2\}$ (the shell of f) is a filter and the set $\{x \in R_2 \mid f(x) \equiv_2 0_2\}$ (the kernel of f) is an ideal.*

Theorem 5.31 *If the shell of a homomorphism f of $P\tau$-algebra is an ultrafilter, then*

1. $f(x) \equiv 1$ *and* $f(y) \equiv 1$ *iff* $f(x \wedge y) = 1$
2. $f(x) \equiv 1$ *or* $f(y) \equiv 1$ *iff* $f(x \vee y) = 1$
3. $f(x) \equiv 0$ *or* $f(y) \equiv 1$ *iff* $f(x \to y) = 1$

Definition 5.34 Let **F** be the set of all formulas of the propositional annotated logic $P\tau$ and f a homomorphism from **F** (considered as a $P\tau$-algebra) into an arbitrary $P\tau$-algebra. We write $f \models \Gamma$, where Γ is a subset of **F**, if for each $A \in \Gamma$, $f(A) \equiv 1$. $\Gamma \models A$ means that for all homeomorphisms f from **F** into an arbitrary $P\tau$-algebra, if $f \models \Gamma$, then $f(A) \equiv 1$.

Based on the above results, we can establish algebraic soundness and completeness of the propositional annotated logic $P\tau$.

Theorem 5.32 (Soundness) *If A is a provable formula of $P\tau$, i.e., $\vdash A$, then $f(A) \equiv 1$ for any homomorphism f from* **F** *(considered as a $P\tau$-algebra) into an arbitrary $P\tau$-algebra.*

To prove completeness, we need the following theorem:

Theorem 5.33 *Let U be an ultrafilter in* **F**. *Then, there is a homomorphism f from* **F** *into* $\mathbf{2} = \{0, 1\}$ *such that the shell of f is U.*

Theorem 5.34 (Completeness) *Let* **F** *be the set of all formulas of the propositional annotated logic $P\tau$ and $A \in$* **F**. *Suppose that $f(A) \equiv 1$ for any homomorphism f from* **F** *(considered as a $P\tau$-algebra) into an arbitrary $P\tau$-algebra. Then, A is a provable formula of $P\tau$, i.e., $\vdash A$.*

Theorem 5.34 gives an alternative completeness result of propositional annotated logics $P\tau$ using Curry algebras $P\tau$. Curry algebras can also be applied to the completeness proof of other paraconsistent logics.

5.5 Formal Issues

There are several important formal issues about annotated logics. *Annotated set theory* can be regarded as a generalization of classical set theory. The most convenient way to study normal structures is to start with a classical set theory, for instance, Zermelo-Fraenkel set theory ZF and to treat them inside ZF. If we proceed this way, then annotated set theory constitutes a natural and immediate extension of fuzzy set theory.

A model theory based annotated predicate logics can be formalized as classical model theory. It is shown that all classical results can be adapted to $Q\tau$. For example, Abe and Akama studied the ultraproduct method for $Q\tau$ in [5]. In fact, $Q\tau$ can provide a unified framework for paraconsistent model theory.

It is interesting to work out a proof theory for annotated logics. Indeed a Hilbert system for annotated logics has been developed, but we need other proof methods for practical applications. For example, a natural deduction formulation was explored in Akama et al. [14] and a tableau formulation was given in Akama et al. [13]. It is also possible to describe sequent calculi for annotated logics. A proof-theoretic study of annotated logics is of help to automated reasoning.

Annotated modal logics can be formalized by extending annotated logics with modal operators. Abe [2] proposed annotated modal logics $S5\tau$ whose modality can be interpreted S5. Akama and Abe [9] investigated annotated modal logics $K\tau$ which corresponds to the normal modal logic.

Annotated logics can be also extended to other modal logics, e.g. temporal, epistemic and deontic logic. Abe and Akama [7] annotated temporal logics for reasoning about inconsistencies in temporal systems.

We can employ annotated logics as a basis for uncertain reasoning. In other words, versions of annotated logics can be formalized as fuzzy, evidential or probabilistic logics. We mentioned these possibilities above.

Work on fuzzy reasoning in annotated logics may be found in Akama et al. [10, 12]. We also attempted to unify annotated and possibilistic logics in Akama and Abe [11].

5.6 Conclusions

We gave a general introduction to annotated logics, which are now considered as important paraconsistent systems. We surveyed propositional and predicate annotated logics with proof and model theory. As an algebraic semantics based on Curry algebras was reviewed. We also make some remarks on formal issues of annotated logics.

We now know many systems of paraconsistent logic, but no systems can provide a unified framework for real applications. Abe and his co-workers established real applications using annotated logics for many years. In this sense, annotated logics can be seen as one of the promising paraconsistent systems. Recent applications of annotated logics to several areas may be found in Abe [4].

Acknowledgments We are grateful to the referee and J.M. Abe for useful comments.

References

1. Abe, J. M.: *On the Foundations of Annotated Logics* (in Portuguese). Ph.D. Thesis, University of São Paulo, Brazil (1992)
2. Abe, J.M.: On annotated modal logics. Mathematica Japonica **40**, 553–56 (1994)
3. Abe, J.M.: Curry algebra $P\tau$. Logique et Analyse **161-162-163**, 5–15 (1998)
4. Abe, J.M. (ed.): Paraconsistent Intelligent Based-Systems. Springer, Heidelberg (2015)
5. Abe, J.M., Akama, S.: Annotated logics $Q\tau$ and ultraproduct. Logique et Analyse **160**, 335–343 (1997) (published in 2000)
6. Abe, J.M., Akama, S.: On some aspects of decidability of annotated systems. In: Arabnia, H.R. (ed.) Proceedings of the International Conference on Artificial Intelligence, vol. II, pp. 789–795. CREA Press (2001)
7. Abe, J.M., Akama, S.: Annotated temporal logics $\Delta\tau$. In: Advances in Artificial Intelligence: Proceedings of IBERAIA-SBIA 2000, LNCS 1952, pp. 217–226. Springer, Berlin (2000)

8. Abe, J.M., Akama, S., Nakamatsu, K.: Introduction to Annotated Logics. Springer, Heidelberg (2015)
9. Akama, S., Abe, J.M.: Many-valued and annotated modal logics. In: Proceedings of the 28th International Symposium on Multiple-Valued Logic, pp. 114–119. Fukuoka (1998)
10. Akama, S., Abe, J.M.: Fuzzy annotated logics. In: Proceedings of IPMU'2000, pp. 504–508. Madrid, Spain (2000)
11. Akama, S., Abe, J.M.: The degree of inconsistency in paraconsistent logics. In: Abe, J.M., da Silva Filho, J.I. (eds.) *Logic, Artificial Intelligence and Robotics*, pp. 13–23. IOS Press, Amsterdam (2001)
12. Akama, S., Abe, J.M., Murai, T.: On the relation of fuzzy and annotated logics. In: Proceedings of ASC'2003, pp. 46–51. Banff, Canada (2003)
13. Akama, S., Abe, J.M., Murai, T.: A tableau formulation of annotated logics. In: Cialdea Mayer, M., Pirri, F. (eds.) Proceedings of TABLEAUX'2003, pp. 1–13. Rome, Italy (2003)
14. Akama, S., Nakamatsu, K., Abe, J.M.: A natural deduction system for annotated predicate logic. In: Knowledge-Based Intelligent Information and Engineering Systems: Proceedings of KES 2007—WIRN 2007, Part II, pp. 861–868. Lecture Notes on Artificial Intelligence, vol. 4693. Springer, Berlin (2007)
15. Anderson, A., Belnap, N.: Entailment: The Logic of Relevance and Necessity I. Princeton University Press, Princeton (1976)
16. Anderson, A., Belnap, N., Dunn, J.: Entailment: The Logic of Relevance and Necessity II. Princeton University Press, Princeton (1992)
17. Batens, D.: Dynamic dialectical logics. In: Priest, G., Routley, R., Norman, J. (eds.) Paraconsistent Logic: Essay on the Inconsistent, pp 187–217. Philosophia Verlag, München (1989)
18. Batens, D.: Inconsistency-adaptive logics and the foundation of non-monotonic logics. Logique et Analyse **145**, 57–94 (1994)
19. Batens, D.: A general characterization of adaptive logics. Logique et Analyse **173–175**, 45–68 (2001)
20. Belnap, N.D.: A useful four-valued logic. In: Dunn, J.M., Epstein, G. (eds.) Modern Uses of Multi-Valued Logic, pp. 8–37. Reidel, Dordrecht (1977)
21. Belnap, N.D.: How a computer should think. In: Ryle, G. (ed.) Contemporary Aspects of Philosophy, pp. 30–55. Oriel Press (1977)
22. Blair, H.A., Subrahmanian, V.S.: Paraconsistent logic programming. Theor. Comput. Sci. **68**, 135–154 (1989)
23. Carnielli, W.A., Coniglio, M.E., Marcos, J.: Logics of formal inconsistency. In: Gabbay, D., Guenthner, F. (eds.) Handbook of Philosophical Logic, 2nd edn, vol. 14, pp. 1–93 Springer, Heidelberg (2007)
24. da Costa, N.C.A.: α-models and the system T and T^*. Notre Dame J. Form. Logic **14**, 443–454 (1974)
25. da Costa, N.C.A.: On the theory of inconsistent formal systems. Notre Dame J. Form. Logic **15**, 497–510 (1974)
26. da Costa, N.C.A., Abe, J.M., Subrahmanian, V.S.: Remarks on annotated logic. Zeitschrift für mathematische Logik und Grundlagen der Mathematik **37**, 561–570 (1991)
27. da Costa, N.C.A., Alves, E.H.: A semantical analysis of the calculi C_n. Notre Dame J. Form. Logic **18**, 621–630 (1977)
28. da Costa, N.C.A., Henschen, L.J., Lu, J.J., Subrahmanian, V.S.: Automatic theorem proving in paraconsistent logics: foundations and implementation. In: Proceedings of the 10th International Conference on Automated Deduction, pp. 72–86, Springer, Berlin (1990)
29. da Costa, N.C.A., Subrahmanian: Paraconsistent logic as a formalism for reasoning about inconsistent knowledge. Artif. Intell. Med. **1**, 167–174 (1989)
30. da Costa, N.C.A., Subrahmanian, V.S., Vago, C.: The paraconsistent logic $P\mathcal{T}$. Zeitschrift für mathematische Logik und Grundlagen der Mathematik **37**, 139–148 (1991)
31. Dunn, J.M.: Relevance logic and entailment. In: Gabbay, D., Gunthner, F. (eds.) Handbook of Philosophical Logic, vol. III, pp. 117–224. Reidel, Dordrecht (1986)

32. Eytan, M.: Tableaux de Smullyan, ensebles de Hintikka et tour ya: un point de vue Algebriquem. Math. Sci. Hum. **48**, 21–27 (1975)
33. Jaśkowski, S.: Propositional calculus for contradictory deductive systems (in Polish). Studia Societatis Scientiarun Torunesis, Sectio A **1**, 55–77 (1948)
34. Jaśkowski, S.: On the discursive conjunction in the propositional calculus for inconsistent deductive systems (in Polish). Studia Societatis Scientiarun Torunesis, Sectio A **8**, 171–172 (1949)
35. Kifer, M., Lozinskii, E.L.: RI: a logic for reasoning with inconsistency. In: Proceedings of LICS4, pp. 253–262 (1989)
36. Kifer, M., Lozinskii, E.L.: A logic for reasoning with inconsistency. J. Autom. Reason. **9**, 179–215 (1992)
37. Kifer, M., Subrahmanian, V.S.: On the expressive power of annotated logic programs. In: Proceedings of the 1989 North American Conference on Logic Programming, pp. 1069–1089 (1989)
38. Kifer, M., Subrahmanian, V.S.: Theory of generalized annotated logic programming. J. Logic Program. **12**, 335–367 (1992)
39. Mortensen, C.: Every quotient algebra for C_1 is trivial. Notre Dame J. Formal Logic **21**, 694–700 (1980)
40. Priest, G., Routley, R., Norman, J. (eds.): Paraconsistent Logic: Essays on the Inconsistent. Philosopia Verlag, München (1989)
41. Priest, G.: Logic of paradox. J. Philos. Logic **8**, 219–241 (1979)
42. Priest, G.: Paraconsistent logic. In: Gabbay, D. Guenthner, F. (eds.) Handbook of Philosophical Logic, 2nd edn., pp. 287–393. Kluwer, Dordrecht (2002)
43. Priest, G.: *In Contradiction: A Study of the Transconsistent*, 2nd edn. Oxford University Press, Oxford (2006)
44. Routley, R., Plumwood, V., Meyer, R.K., Brady, R.: *Relevant Logics and Their Rivals*, vol. 1. Ridgeview, Atascadero (1982)
45. Subrahmanian, V.: On the semantics of quantitative logic programs. In: Proceedings of the 4th IEEE Symposium on Logic Programming, pp. 173–182 (1987)