ISSN 1870-4069

Béziau's SP3A Logic and Logic Programming

Jaime Díaz[1], José Luis Carballido[1], Mauricio Osorio[2]

[1] Benemérita Universidad Autónoma de Puebla,
Facultad de Ciencias de la Computación, Mexico

[2] Universidad de las Américas Puebla,
Departamento de Actuaría, Física y Matemáticas, Mexico

{jdiazalt27, jlcarballido7, osoriomauri}@gmail.com

Abstract. We present some three-valued paraconsistent logics, in particular SP3A logic which has been recently introduced as a genuine paraconsistent logic. After discussing the relevance of some rules in classical logic, we motivate the need for paraconsistency. Then we mention areas of applications of non-monotonic reasoning in terms of programming semantics, and mention the relationship that exists between these semantics and logics. In particular we show that logic SP3A belongs to the family of D-elemental logics; these logics can characterize one of the semantics useful in the area of non-monotonic reasoning.

Keywords. Logic programming, paraconsistency, 3-valued logic.

1 Introduction

Consistency is a property that comes from complying that $x \wedge \neg x$ is always false. This means that when a formal system can never produce two or more formulations that contradict each other, then we say that such system is *consistent*. Classical logic is consistent and holds many other desirable properties: de Morgan laws, distributivity, among others and is the right choice whenever a solid and consistent theoretical foundation is needed. Therefore, mathematics, physics, chemistry and engineering take a hold on the basis of classical logic since these disciplines work with instances that generally do not allow contradictions.

This view of the necessity of consistence was strongly defended by Aristotle, and is called *the law on non contradiction (LNC)* (presocratic philosophers did not consider consistency as a matter of necessity) [11]. Logical systems that comply with consistency are called *classical* or *aristotelian*; in aristotelian systems, if we accept a formula that does not comply with the LNC, *explosivity* occurs: the logical system explodes into an infinity of propositions that can be proved as true. This can be expressed as $x \wedge \neg x \rightarrow y$ However, the LNC has not been formally proved yet, and even Aristotle failed trying to show the necessity of its compliance [11]. On the other hand, not only presocratics believed that consistency in a classical way is not necesary for a formal system to be valid: as we will see later, logics that do not obey LNC have been developing since the beginning of XXth century. Non classical logics arise then as valid logics even when they do not hold the LNC, and arise too as an answer to situations where contradiction is an inherent feature.

In next section (section 2) we briefly review formal systems and their structure. Section 3, is about paraconsistent logics; in particular we present SP3A and SP3B logics. Section 4 is a brief review of logic programming, where we review some transformations of logic programs in order to avoid inner contradictions. Finally, in the last section we present D-elemental logics and a demonstration that SP3A logic is D-elemental.

2 On Formal Systems, Semantics and Axiomatics

As da Costa remarks in [5], it is difficult to give a precise definition to *semantics*, specially when working with non-classical logics. However, up until now, and given that our pourpose is not a deep philosophical study of semantics, we will understand it as da Costa mentions: *"When first proposed in the fields of logic and formal sciences, the term «semantics» send to present a clear sense. It was supposed to denote that part of an analysis of a language concerned with the determination of the meanings of its (well formed) expressions"* [5]. In this sense, we will then consider that a logic can be defined and analysed when its connectives are defined by presenting their behavior through a truth-table, and well formed formulas (wfs) are presented and evaluated by this same way. This means that semantics allows us to determine if a statement is a tautology, a contradiction, or which is its particular evaluation for any specific interpretation of its variables; moreover, it helps us to determine wether two statements are equivalent to one another. Going back to da Costas's definition of semantics, it must be noted that it implies the determination of the meaning of a wfs. Then, as the semantic approach leads us to elaborate truth-tables that evaluate wfs to defined truth values, at the end we will have a bunch of truth values as the meaning of that specific wfs. Then, it is necessary to understand the meaning of all the truth values in the domain of that particular logic. In classical logic they are ($0 = False$ and $1 = True$), but when it comes to many-valued logics, the matter is not so obvious. Later we will discus about this particular issue.

Mendelson on the other hand, illustrates the axiomatic method by defining a formal theory (a formal system). As Mendelson claims, a formal theory needs 4 conditions to be satisfied:

1. A set of countable symbols to form expressions. An expression is a finite sequence of these symbols.
2. A set of countable expressions that will be called well-formed formulas, and a method to determine if an expression is a wfs.
3. A set of wfs called *axioms*.
4. A finite set of rules of relation between wfs (rules of inference) [8].

Using these rules of inference over axioms, new valid wfs can be obtained. If an obtained expression is a tautology, then it is called a *theorem*.

By this method, we can generate new theorems, and prove things such as logical equivalence or dependence among expressions.

Semantic and axiomatic approaches must not be considered as opposite but as complementary; it is natural to use both in order to perform proofs and calculations.

3 Paraconsistent Logics

In this section we will review some background of paraconsistent logics: What is paraconsistency, forerunners of paraconsistet logic and finally, we will present two paraconsistent logics proposed by Jean-Yves Béziau.

3.1 Non-classical Logics: 3-valued and Paraconsistent Logics

As stated in [3], a *logic* is a set of wfs that satisfies two properties:

1. The set is closed under Modus Ponens: if A and $A \to B$ are in the set, then B is in the set too.
2. The set is closed under substitution: if the formula A is in the logic, then any other formula obtained by replacing all the ocurrences of an atom b in A with other formula B is in the logic too.

Classical logic is the most well known logic, and holds many well defined characteristics; it is 2-valued, obeys LNC, and double negation (the negation of the negation of a formula is equivalent to the original formula $\neg\neg x \to x$); in general, classification of logics is not a precise matter, and it is difficult to claim that a logic is classified in a particular way. However, for us, non-classicism means at least one of two characteristics:

- The logic has 3 or more values.
- The logic does not obey the LNC.

In this sense, our object of interest (SP3A logic) holds both of them, so it will be called a non-classical logic.

Paraconsistency:

There is a group of non-classical logics that are called *paraconsistent logics*. da Costa is one of the main initiators of paraconsistent logics, and his original definition was *"a one-place operator ν is paraconsistent if there is a formula a such that the theory $a, \eta a$ is non-trivial"*, that is, a and ηa are both true simultaneously, with the constraint that this operator should obey all the properties of classical negation (though it is not clear which properties are these) [5].

According to Béziau , *"Paraconsistent logic can be considered a bunch of logical systems in which there is a connective which does not obey the principle of non-contradiction; such a connective is usually called a paraconsistent negation and the main problem is to know if it is legitimate to call such an operator a negation"* [2].

These paraconsistent logics make it possible to deal with contradictions without our formal system exploding into triviality. This property is useful when working with contradictory information (as in non-monotonic reasoning) and makes paraconsistent logics a powerful tool when we need to deal with contradictions; in this work we will focus in just one field of application: Artificial Intelligence.

3-valued logics and the interpretetion of the third value

Among non-classical logics, there is a group of logics that are called *multivalued*; these multivalued logics hold a dominion D with more than two truth values. Logically, there are three-valued logics in which $D = \{0, 1, 2\}$, and where 0 and 2 usually behave classically just as classical 0 and 1 do; but in these logics, the value 1 must be interpreted in a particular way. There are several interpretations for this value, and the interpretation must match with the logic semantics and this interpretation makes the particular logic useful for specific areas of study.
In [6], Coste-Marquis considers multi-valued logics as useful when working with contradictory pieces of information (when working with agents for example); more specifically 3-valued logics, and where the third value (1) is given a specific interpretation: *"Proved both True and False"*. However, this approach does not guarantee that its inferences are consistent. Later we will introduce SP3A and SP3B logics. These logics need an interpretation for their third values. We will propose an interpretation for SP3A.
Although in [6] a particular interpretation for the third value in 3-valued logics is "both true and false", there can be many different ways to understand it. In [4], Ciucci and Dubois mention some of them: possible, unknown, undefined, half-true, irrelevant, inconsistent. These interpretations can be classified in two types: *ontological* (undefined, half-true, irrelevant) and *epistemic* (possible, unknown). Ontological values make reference to a situation where the nature of the third value is not questioned, but understood as an intrinsic feature of the expression; epistemic values on the other hand, are values whose state will eventually change into 0 or 2 in the future.

3.2 Forerunners of Paraconsistency: Łukasiewicz 3-valued Logic, Vasili'év Logic and G_3 Logic

History of paraconsistent logics is long yet almost unknown for most people; Priest states that presocratic philosophers where familiar with logical systems that did not obey the LNC, and it was Christianism that took Aristotelian theories, and the LNC as a dogma for centuries [11]. As a result, classical logic was the only one that developed along this period. In 1910 Jan Łukasiewicz published the book *On the Principle of Contradiction in Aristotle*. This book studies Aristotle's LNC, and concludes that it can not be proved in the sense that *every contradiction is false*: even if Aristotle's arguments were proved, it would prove only that *some contradictions are not true*. Łukasiewicz gives then what he claims is *the only strict and formal proof* for the LNC: the only way to prove it is to assume that contradictory objects are not objects at all, it means they are *nothing*, instead of *something*; this means that anything that is *something* and not *nothing*, does not contain contradictory properties [12].

Łukasiewicz was one of the forerunners on non-classical logics; as he studied the LNC and found out there was no way to prove it valid for every contradiction, he set the precedent for paraconsistent logics, and defined the family of logics $Ł_\omega$.
Vasili'év is also considered a precursor of paraconsistency. Inspired by Lobatchevsky's non-Euclidian geometry, he had some ideas about what he called *imaginary logic* [5]. Łukasiewicz and Vasili'év ignited the development of paraconsistency.

Finally, we will mention a logic defined by Göedel. This logic belongs to the family of multivalued logics G_i, and is the 3-valued logic G_3. The relevance that this logic has for our work is that SP3A logic has no implication connective defined in it and given that we will need it later, we will use G_3 native implication for it.

3.3 Béziau's Paraconsistent Logics SP3A and SP3B

In this work, we present the analysis of some features of SP3A logic. This is a 3-valued paraconsistent logic proposed by Jean-Yves Béziau in *Two Genuine 3-Valued Paraconsistent Logics* [1], where Béziau presents SP3A and SP3B . Each of these 3-valued logics have only three primitive connectives: negation, conjunction and disjunction. In tables 1 and 2, we respectively present the definition of SP3A and SP3B connectives:

Table 1. Truth tables of connectives ∧, ∨, and ¬ in $SP3A$.

$\wedge$	0	1	2
0	0	0	0
1	0	1	2
2	0	2	2

$\vee$	0	1	2
0	0	1	2
1	1	1	2
2	2	2	2

x	$\neg x$
0	2
1	2
2	0

Table 2. Truth tables of connectives ∧, ∨, and ¬ in $SP3B$.

$\wedge$	0	1	2
0	0	0	0
1	0	2	1
2	0	1	2

$\vee$	0	1	2
0	0	1	2
1	1	1	2
2	2	2	2

x	$\neg x$
0	2
1	1
2	0

According to the definition of paraconsistent logic given by Béziau , SP3A would be a paraconsistent logic since its negation does not obey the LNC. This can be seen in Table 3

Table 3. Law of Non Contradiction in $SP3A$ and $SP3B$.

x	$\neg x$	$x \wedge \neg x$	$\neg(x \wedge \neg x)$
0	2	0	2
1	2	2	0
2	0	0	2

x	$\neg x$	$x \wedge \neg x$	$\neg x \wedge \neg x$
0	2	0	2
1	1	1	1
2	0	0	2

As can be seen, $\neg(x \wedge \neg x)$ is not a tautology in neither of these logics. That is why Béziau claims these are paraconsistent logics.

Béziau presents his logics and analyses some of their behaviors, mainly in properties related to negation, such as de Morgan laws and double negation; Béziau shows too that substitution theorem does not hold in neither of these logics.

4 Logic Programming

In *"Paraconsistent Logic in a Historical Perspective"* [5], the authors state that the future of Paraconsistent Logic lays in some research lines such as:

- To develop a Paraconsistent Model Theory.
- To develop a paraconsistent Set Theory.
- To develop a Paraconsistent Mathematics.
- To develop further applications to Computer Science, Artificial Intelligence, Law, Everyday Life...

It is in particular the last item in the list above the matter of interest for this work: when working with Logic Programs, Belief Databases, interaction with human beings, sensor fusion, it is common that inconsistencies appear, collapsing the system. There are techniques to avoid it, and using paraconsistent logics to treat the information is one of them. In this section we explore some background of logic programming and belief revision.

Logic programs

Logic programs are formed by 1 or more logic formulas; these programs are designed to make controlled logic inferences in order to demonstrate theorems automatically [3].

According to Schlipf, a *literal* is an atomic formula $R(t_1, t_2...t_n)$ (a *positive* literal) or a negated atomic formula $\neg R(t_1, t_2...t_n)$ (a *negative* literal).

A *logic program* is a finite, or countably infinite, set of *rules*, (implicitly universally quantified) formulas of the form $a \leftarrow b_1 \wedge b_2 \wedge ... \wedge b_n$, where a is a positive literal and the b_i s' are all literals -positive or negative- [13].

Note that in [13], the head of a rule contains just a single literal; however we will use a more general approach where rules are expressions in the form $A \leftarrow B$ where $A = a_1 \vee a_2 \vee ... \vee a_k$ and $B = b_1 \wedge b_2 \wedge ... \wedge b_n \wedge \neg b_{n+1} \wedge ... \wedge \neg b_{n+m}$, with $k \geq 1$ and where $a_i' s$ and $b_i' s$ are atomic formulas [3].

According to Carballido, we can say:

Definition 1. *A logic program (LP) is a* theory*, this is, a set of formulas; a* class of Logic Programs *is a set of LP that satisfies certain property or syntactical limitation. [3].*

Definition 2. *Rules used in Logic Programming hold the following structure: $A \leftarrow B$, where A is called* head of the rule *and B is called* body of the rule. *[3].*

Note that $A \leftarrow B$ is usually used as an alternate form to $B \rightarrow A$ ($\leftarrow$ is a kind of implication [13]).

In order to abbreviate notation, it is common to write a rule in the form: $A \leftarrow B^+ \wedge \neg B^-$, where $B^+ = \{b_1, b_2, ..., b_n\}$ and $B^- = \{b_{n+1}, b_{n+2}, ..., b_{n+m}\}$

Types of programs and rules

Any rule $A \leftarrow B$, where $A = a_1 \vee a2 \vee ... \vee a_k$, $k \geq 1$, and $B = b_1 \wedge b_2 \wedge b_3 \wedge ... \wedge \neg b_n \wedge \neg b_{n+1} \wedge ... \wedge \neg b_{n+m}$, with $n \geq 0$ and $m \geq 0$ is called a *clause or disjunctive rule*. In case that $n = 1$ we will call it a *Normal Clause*. If the clause has no negated atoms, then will call it *Positive*. A program which all of its rules are disjunctive is called a *Disjunctive LP*, and a program which all of its rules are normal is called a *Normal LP*. The size of a disjunctive clause is defined by $k + n + m$ [3]. If there is a program in the form $A \leftarrow$, then it is called a *fact*; an expression in the form $\leftarrow B$ is called a *restriction*.

Stable Model Semantics

In order to explain what the Stable Model Semantics is, we will give two definitions:

Definition 3. *If a logic is stronger than intuitionistic logic, and weaker than classical logic is called* intermediate logic. *An intermediate logic is called* propper *if it is strictly contained in classical logic. Intuitionistic logic is an intermediate logic too [3] [10].*

One intermediate logic to mention is G_3; No one of the intermediate logics is paraconsistent [3].

Definition 4. *If M is a set of atoms of a logical program P, M is a classical model of P, and M is minimal among the classical models of P, then we say that M is a* minimal model *of P.*

Example

Let P be the LP shown:

$a \leftarrow b \wedge \neg c$
$b \leftarrow \neg a$
$b \leftarrow c$

The classical models for P are: $\{a\}$, $\{c, b\}$, $\{a, b\}$, $\{a, b, c\}$.
Minimal classical models are: $\{a\}$, $\{c, b\}$.

In [9], Osorio defines Stable Model Semantics as shown in the following definition:

Definition 5. *Given a disjunctive program P, for any set M of atoms in P, P^M is the program obtained from P by removing:*

1. *Each clause that contains at least one negative literal $\neg b$ in its body, whith b $\in M$.*
2. *All the negative literals in the bodies of remaining clauses.*

Example

Taking the program from the previous example, and considering $M_1 = \{a\}$ and according to the definition of stable, we get P^{M_1} as:

$a \leftarrow b$
$b \leftarrow c$

Models of P^{M_1} are: $\emptyset$, $\{a\}$, $\{a,b\}$ and $\{a,b,c\}$.
Proposing $M_2 = \{b,c\}$, and we can obtain P^{M_2} from applying 5 as:

$b \leftarrow$
$b \leftarrow c$

Models of P^{M_2} are: $\{b\}$ and $\{b,c\}$.

Definition 6. *It is a kwnown fact that any possitive program will always contain at least* 1 *classical minimal model. If M is one of the minimal models of P_M, then we say that M* is a stable model of P *[3] [9].*

Example

From preceding 2 *examples we already know program P, program P^{M_1}, P^{M_2} and the models for each of them. $\{a\} = M$ but it is not a Stable Model because $\{a\}$ is not a minimal model of P^{M_1} ($\emptyset \in \{a\}$).*

Analyzing P^{M_2}, its minimal model is $\{b\} \neq M_2$. Neither M_1 nor M_2 are stable models of P.

In [10], the author shows that the stable semantics can be characterized by modeling in intermadiate logics. We present next another semantics that can also be applied in non-monotonic reasoning, and can be characterized in terms of paraconsistent logics, in particular D-elemental logics. This semantics is called p-stable semantics and offers an alternative to the stable semantics [3].

P-stable Model Semantics

Next, we define another transformation that is helpful to define an alternative semantics to the stable model semantics.

Definition 7. *Let P be a program and M a set of atoms in P. Then, we define $RED(P, M) := \{a \leftarrow B^+ \wedge \neg(B^- \bigcap M) \mid a \leftarrow B^+ \wedge \neg B^- \in P\}$.*

Definition 8. *Let P be a normal program and M a set of atoms in P, then, if $RED(P, M) \Vdash_C M$, we say that M is a p-stable model of P. Where the symbol $\Vdash_C$ means that the formulas on the right can be proved from those on the left in classical logic.*

Example

Continuing with the example, , if we take $M_1 = \{a\}$, then we get $RED(P, M_1)$:

$a \leftarrow b$
$b \leftarrow \neg a$
$b \leftarrow c$

By following the next equivalences: $\neg a \rightarrow b \equiv \neg b \rightarrow \neg\neg a \equiv \neg b \rightarrow a$ so we have $(b \rightarrow a) \wedge (\neg b \rightarrow a)$. From this formula it follows a, and then, M_1 is a p-stable model of P.

4.1 Treating Inconsistency in Artificial Intelligence

When a system interacts with the world, there is a flux of information to take in account; the world changes constantly and new information is generated every instant. Our system has then the necessity to constantly update the information in its data bases. But it is common that new and previous information are opposed to one another. Even more, it may occur that given a set of sensors, some of them give contradictory information at the same instant in time (in sensor fusion for example). If the system interacts with people, the possibility of inconsistent information is even higher: humans are naturally so complex and behave in contradictory ways: a person changes his mind from one moment to another, judges situations in a subjective way and frequently acts in a different way than he thinks or for every enviroment acts in a special manner. If such a system is based on classical logic, these contradictions lead to inconsistencies that could make the system explode so that $(x \wedge \neg x) \rightarrow y$, and preventing contradictions becomes a matter of major importance. But if we have a way to deal with contradictions withouth our system exploding, then the problematic situation loses strenghth.

When working with contradictory information there are several approaches that allow us to avoid the trivialization. This is a fundamental problem in A.I.; The problem can be treated by belief revision, belief merging, reasoning from preferred subsets, purification, paraconsistency, etc. The existence of such a wide spectrum of approaches can be explained by the fact that paraconsistency can be achieved in various ways; the particular situation stablishes the needs to be fulfilled and the way to follow in order to achieve this paraconsistency[6].

Paraconsistency taken in a strict sense allows us to deal with inconcistency directly, while the other approaches require some extra-logical information to avoid trivialization; the limitation in these cases is in the fact that this extra information may be too poor or too sophisticated[6].

Belief Revision

Human beings are always changing our minds: new information gets to us and modifies our considerations and knowledge about the world: changes our beliefs. When working with *agents* in Artificial Intelligence leads us to a similar scenery: an *agent* is defined by Russel and Norvind as *"any entity capable of perceiving and acting on the world"* (cited in [7]). When an agent interacts with an environment, fresh information is arriving all the time making it necessary to it to change its «beliefs» (Belief is a relation between an agent and a proposition; however it is difficult to restrict which relations are beliefs and which are not [7]). *Belief Revision* is the discipline that studies the rationality of belief change in agents [7].

Moretto mentions as an example a few agents working in a determined environment.

5 SP3A logic is D-elemental

This last section is dedicated to prove that SP3A logic is D-elemental; definition 9 tells us what does it mean that a logic is D-elemental:

Definition 9. *A multi-valued logic E is* D-Elemental *if its domain D contains three special elements 0, 1 and t, that satisfy the following properties [3]:*

1. *t is a designated value, and 0 is undesignated*
2. *The value assigned to $1 \rightarrow 0$ s not designated*
3. *Connectives $\wedge$ and $\vee$ are commutative and associative*
4. *For every value of x, $0 \wedge x = 0$ and $0 \rightarrow x \in \{1, t\}$*
5. *For every special element x, $1 \vee x \in \{1, t\}$*
6. *For every special element x, $t \vee x \in \{1, t\}$*
7. *Fragment $\{0, t\}$ matches classical logic (for $\wedge$, $\vee$, $\rightarrow$ and $\neg$)*
8. *Fragment $\{0, t\}$ is closed respect to $\wedge$ and $\rightarrow$ connectives*
9. *The value assigned to negation of 1 is an element in $\{1, t\}$*
10. *E logic lies between C_ω and C, where C is classical logic. That is $C_\omega \subset E \subset C$*

In the following lines we will prove that SP3A logic is $D - elemental$

The G_3 implication

The logic SP3A does not include any implication as a primitive connective; however, it is necessary to have an implication in SP3A in order to prove it is a $D - elemental$ logic. In order to have an implication for SP3A , now, we are going to define G_3 implication in SP3A terms. G_3 implication is defined as shown in 10

Definition 10. *G_3 implication is defined by:*

$$x \rightarrow y = \begin{cases} 1 : x \leq y \\ y : x > y \end{cases}$$

Within SP3A , G_3 implication will be denoted by the symbol "$\rightarrow_G$", and it is expressed in terms of the SP3A connectivesby:

$$x \rightarrow_G := (\neg x \vee y) \wedge (\neg(x \vee (x \wedge \neg x)) \vee y)$$

The Possitive logic

The *Possitive logic* (Pos) is defined by a set of 8 axiomatic schemes; these schemes are listed below:

For every wfs α, β and γ [3]:

Pos1 $\alpha \rightarrow (\beta \rightarrow \alpha)$
Pos2 $(\alpha \rightarrow (\beta \rightarrow \gamma)) \rightarrow ((\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow \gamma))$
Pos3 $\alpha \wedge \beta \rightarrow \alpha$
Pos4 $\alpha \wedge \beta \rightarrow \beta$
Pos5 $\alpha \rightarrow (\beta \rightarrow (\alpha \wedge \beta))$
Pos6 $\alpha \rightarrow (\alpha \vee \beta)$
Pos7 $\beta \rightarrow (\alpha \vee \beta)$
Pos8 $(\alpha \rightarrow \gamma) \rightarrow ((\beta \rightarrow \gamma) \rightarrow (\alpha \vee \beta \rightarrow \gamma))$

It can be easily proved that SP3A obeys all these axiom schemes using a semantical approach.

The C_ω logic

C_ω logic was proposed by daCosta ; it is defined by the axiom set of Pos, plus $C_\omega 1$ and $C_\omega 2$.

Let α be a well formed formula in C_ω, then

$C_\omega 1$	$\alpha \vee \neg\alpha$
$C_\omega 2$	$\neg\neg\alpha \to \alpha$

C_ω is a *minimal paraconsistent logic* which means that any other logic E, in which $\neg(A \wedge \neg A)$ is not a theorem must contain every C_ω theorem [3]. Then, as a consequence of this affirmation we can state the next lemma:

Lemma 1. *SP3A logic contains all the theorems in C_ω logic: $C_\omega \subseteq SP3A$*

Proof: As shown in table 3, $\neg(A \wedge \neg A)$ is not a theorem in SP3A. Then, SP3A must contain every C_ω theorem.

SP3A logic is *D-elemental*

Now we proceed to verify that SP3A logic obeys the 10 points that define a *D-elemental* logic:

1. Dominion of variables in SP3A is $D = \{0, 1, 2\}$; in this sense, SP3A logic satisfies this point: 0 and 1 are in its domain, and there is a designated value $t = 2$ while 0 is undesignated
2. In SP3A , $1 \to_G 0 = 0$; 0 is not designated
3. In SP3A , connectives $\wedge$ and $\vee$ are commutative and associative; it can be proved easily using truth tables
4. For any x in the domain of SP3A , $0 \wedge x = 0$; $0 \to_G x \in \{1, 2\}$. This occurs because $x \in \{0, 1, 2\}$ and in every case $0 \leq x$ and according to the definition of the G_3 implication, in these cases the evaluation is 1
5. In SP3A , $\vee(x, y) = max(x, y)$. Then, $1 \vee x \in \{1, 2\}$ since $x \in \{0, 1, 2\}$
6. As in SP3A $\vee(x, y) = max(x, y)$, then $t \vee x = 2 \vee x = 2$, and $2 \in \{0, 1, 2\}$
7. In SP3A , fragment $\{0, t\} = \{0, 2\}$ behaves classically as can be seen in table 4
8. let $x \in \{1, 2\}$, then $1 \wedge x \in \{1, 2\}$ and $2 \wedge x = 2$. It occurs too that $1 \to_G x = 2$ and $2 \to_G x \in \{1, 2\}$. Then we can say that in SP3A the fragment $\{1, 2\}$ is closed for connectives $\wedge$ and $\to_G$
9. $\neg 1 = 2$, $2 \in \{1, 2\}$
10. As stated in lemma 1, $C_\omega \subseteq SP3A$; all theorems in SP3A can be proved in C. That is: $C_\omega \subseteq SP3A \subseteq C$

Theorem 1. *SP3A is a* D-elemental *logic.*

Proof: as shown previously.

Paraconsistent logics and logic programming

Finally, we will mention a theorem as stated and proved in [3]:

Theorem 2. *Let P be a disjunctive program. Let M be a set of atoms in P, and let E be a D-elemental logic. $P \bigcup \neg\widetilde{M} \vdash_E M$ iff $RED(P, M) \vdash_E M$.*

Table 4. Truth tables of connectives $\wedge$, $\vee$, $\neg$ and $\rightarrow_G$ for the fragment $\{0, 1, 2\}$ in $SP3A$.

$\wedge$	0	2
0	0	0
2	0	2

$\vee$	0	2
0	0	2
2	2	2

x	$\neg x$
0	2
2	0

$\rightarrow_G$	0	2
0	0	2
2	2	2

References

1. Béziau, J.Y.: Two genuine 3-valued paraconsistent logics. In: Towards Paraconsistent Engineering, pp. 35–47. Springer (2016)
2. Béziau, J.Y.: What is paraconsistent logic? In: In Batens et al. [6. pp. 95–111. Press (2000)
3. Carballido, J.C.: Fundamentos Matemáticos de la Semántica P-estable en Programación Lógica. Ph.D. thesis, Benemérita Universidad Autónoma de Puebla (2009)
4. Ciucci, D., Dubois, D.: A map of dependencies among three-valued logics. Information Sciences 250, 162–177 (2013), `http://www.sciencedirect.com/science/article/pii/S0020025513004775`
5. da Costa N.C.A., B.J., O., B.: Aspects of paraconsistent logic (1995)
6. Coste-Marquis, S., Marquis, P.: Recovering consistency by forgetting inconsistency. Lecture Notes in Computer Science 5293 (2008)
7. M., M.: Belief Revision in Non-Classical Logics. Springer (2013)
8. Mendelson, E.: Introduction to Mathematical Logic. Chapman and Hall (1997)
9. Osorio M., Navarro J.A., A.J., V., B.: Logics with common weak completions. Journal of Logic and Computation 16 (2006)
10. Pearce, D.: Stable inference as intuitionistic validity. The Journal of Logic Programming 38, 79–91 (1999)
11. Priest, G.: Why it's irrational to believe in consistency. In: Brogaard, B., Smith, B. (eds.) Proc. 23 rd International Wittgenstein Symposium. pp. 284–293 (2001)
12. Priest, G.: Torn by reason: łukasiewicz on the principle of non contradiction. Early Analytic Philosophy - New Perspectives on the Tradition (2016)
13. Schlipf, J.: Formalizing a logic for logic programming. Annals of Mathematics and Artificial Intelligence 5, 279–302 (1992)