On Contradictions and their Propagation in Paraconsistent Logic Programs

João Alcântara, Carlos Viegas Damásio and Luís Moniz Pereira

Centro de Inteligência Artificial (CENTRIA)
Departamento de Informática, Universidade Nova de Lisboa
2829-516 Caparica, Portugal. e-mail: jfla|cd|lmp@di.fct.unl.pt

Abstract. Paraconsistent Logic Programs are a powerful mathematical formalism for dealing simultaneously with default, paraconsistent, and uncertainty reasoning forms. Disclosing recent results, we show how their semantics, *Paraconsistent Well-Founded Models* (WFM^p), deal with inconsistent information and its propagation through program rules. In doing that we emphasize another key role the *Coherence Principle* plays in our semantics, and generalise results presented in [7]. By means of an illustrative example, we argue that not only should one extract paraconsistent conclusions when facing inconsistent information, but one may also obtain a precise clue as to which information depends on inconsistency.

1 Introduction

As noticed in [5], contradictions can be considered under the mantle of many points of view: as a consequence of the only correct description of a contradictory world, as a temporary state of our knowledge, as the outcome of a particular language which we have chosen to describe the world, as the result of conflicting observational criteria, as the superposition of world-views, or as the result from the best theories available at a given moment. Because of the presumable common presence of contradictory theories, paraconsistent logics have been proposed to support them in the task of significantly drawing conclusions from them.

The Logic Programming community is not apathetic about the importance of paraconsistency in enlarging our understanding of the common sense reasoning (see [7] for a survey). Aiming at the characterisation of a general framework for logic programs, we have introduced Paraconsistent Logic Programs [1, 2], interpreted by the *Paraconsistent Well-Founded Model* (WFM^p). Such programs generalise Antitonic Logic Programs [8] by ushering in concepts to cope with explicit negation and default negation, and certifying that default negation complies with the Coherence Principle, stating that explicit negation entails default negation. Paraconsistent Logic Programs are general enough to capture many forms of uncertain reasoning studied in Logic Programming [1].

Following results unearthed in the semantics $WFSX_p$ [6], we claim that besides reasoning paraconsistently when facing inconsistencies, we should keep track of which information depends on them. Irrespectively of how we view inconsistencies, to be aware that a conclusion whether or not depends on them is vital to avoid misleading reasonings. Accepting that politics is a field endemic of contradictions, we illustrate our claim by resorting to recently occurred events in Portuguese public life:

On December 10th 2002, workers were summoned by the main Portuguese trade union confederation to a general strike against reforms in the labour legislation intended by the government. A massive popular support for or against reforms is decisive for its acceptance or refusal. Believing faithfully that the Portuguese parliament thoroughly accedes to the expectancy of the society, we can translate the above situation into the following Paraconsistent Logic Program rules[1]

$$\begin{cases} accept \leftarrow \otimes_k \ <1,\,0> \ for\_reforms \\ \neg accept \leftarrow \otimes_k \ <1,\,0> \ against\_reforms \end{cases}$$

Allegedly, these reforms will be used to cut down the budget deficit, which corresponds to the rule $cut\_deficit \leftarrow \otimes_k <1,\,0> accept$.

The strike finished, both sides involved in the quarrel hurried to disclose their figures. With the seriousness recommended by the liturgy of his position, the Minister of Labour revealed that only 13% of the workers had supported the strike [9]; almost at the same time, with the same seriousness, the trade union's confederation president reckoned that 85% had striked [15]. Gathering the latter information along with the previous rules, and assuming that everyone working on this day was favourable to the proposed reforms, we have the Paraconsistent Logic Program P below:

$$\begin{array}{ll} accept \leftarrow \otimes_k \ <1,\,0> \ for\_reforms & cut\_deficit \leftarrow \otimes_k \ <1,\,0> \ accept \\ \neg accept \leftarrow \otimes_k \ <1,\,0> \ against\_reforms & for\_reforms \leftarrow <0.87,\,0.13> \\ against\_reforms \leftarrow <0.85,\,0.15> & \end{array}$$

Employing a bilattice defined over the real unit interval to represent the truth-values, we obtain by the WFM^p a (degree of) inconsistency in which 87% of the workers accept the reforms, while 85% do not. The polemic point is concerned with *cut_deficit*. In conformity in the WFM_p, *cut_deficit* is assessed at $< 0.87, 0 >$, i.e., there is a degree of confidence of 87% in that the budget deficit will be decreased. So *cut_deficit* has a consistent value, but grounded on the inconsistent conclusion of *accept*. What can we do? If we just accept its value, we will favour government policy; in turn, if we refuse this result because it depends on inconsistency, we will favour trade union position; finally, if we simply reject the possibility of contradictions existing in a real-world situation, we will be accused, by raging paraconsistent logicians, of supporting only orthodox forms or reasoning. Now imagine that the cut in budget deficit is fundamental for friendly conviviality in the European Community. Perhaps they themselves will not feel comfortable in concluding *cut_deficit* on the basis of inconsistent premisses.

A solution satisfying all parties involved would allow concluding *cut_deficit*, and concomitantly to make clear that it reposes on the taint of inconsistency. To guarantee that the WFM^p provides this solution is the central point of the

[1] We use $\otimes_k$ to denote the meet operator under knowledge ordering in a bilattice in prefix notation. See the definition of PLP algebra $\mathfrak{R}$ in Example 1 for technical details.

present paper. With this information on hand, the end-user (in this case, the European Community) is afforded worthwhile information in order to decide what to do: be it to accept it, reject it, or to accept or reject it under conditional terms, because of its dependence on inconsistency.

This paper is structured as follows: in section 2, we present Paraconsistent Logic Programs and their semantics. In section 3, we exhibit results concerning how we can identify propositional symbols which are inconsistent or which depend on inconsistency. Finally, we draw conclusions and refer to related work.

2 Paraconsistent Logic Programs

In the framework of Antitonic Logic Programs [8], rules are constituted either by arbitrary isotonic[2] or antitonic body functions, and by propositional symbols in the head. Because of their arbitrary isotonic and antitonic operators over a complete lattice, these programs pave the way to combine and integrate into a single framework several forms of reasoning, such as fuzzy, probabilistic, uncertain, and paraconsistent ones. Following another path, Lakshmanan and Sadri [16] proposed a framework for modeling uncertainty, where both belief and doubt are explicitly incorporated by using a bilattice to represent truth-values.

Definition 1 (Bilattice). *A bilattice is a structure* $\mathfrak{B} = \langle B, \leq_t, \leq_k \rangle$ *where* B *is a nonempty set, and* $\langle B, \leq_k \rangle$ *and* $\langle B, \leq_t \rangle$ *are both lattices with a bottom and a top element. We shall use* $\otimes_k$ *and* $\oplus_k$ *respectively for denote the meet and join operations that correspond to* $\leq_k$*, and* $\otimes_t$ *and* $\oplus_t$ *for those that correspond to* $\leq_t$.

If both $\langle B, \leq_k \rangle$ and $\langle B, \leq_t \rangle$ are complete lattices, i.e. their respective least upper bound and greatest lower bound exist for arbitrary subsets of the lattice (not only for finite subsets), we say that the bilattice is *complete*. The partial order $\leq_k$ is intended to represent the knowledge or information order, and $\leq_t$, the truth order. Originally defined by Ginsberg [14], bilattices are particularly fitted to characterise important operators found in logic programming. In order to introduce a negation operator ($\neg$) in a bilattice, besides satisfying the double negation property ($\neg\neg a = a$), negation must invert the notion of truth, whilst preserving the knowledge order. As for the conflation operator ($-$), besides satisfying the double conflation property ($- - a = a$), conflation must invert the notion of knowledge, whilst preserving the truth order. In a bilattice $\mathfrak{B} = \langle B, \leq_t, \leq_k \rangle$ admitting that $\neg$ and $-$ commute[3], we can define the default negation operator *not* $: B \to B$, as *not* $a = -\neg a$.

Furthermore, the notion of conflation is used by Fitting [11] to define important relations over the bilattice. In particular, this notion authorise us to say

[2] A function is isotonic (antitonic) iff the value of the function increases (decreases) when we increase any argument while the remaining arguments are kept fixed.

[3] In a bilattice $\mathfrak{B} = \langle B, \leq_t, \leq_k \rangle$, we say $\neg$ and $-$ *commute* if $\neg - A = -\neg A$ for all $A \in B$.

that in a bilattice $\mathfrak{B} = \langle B, \leq_t, \leq_k \rangle$ with conflation, for $A \in B$, A is inconsistent if $A \not\leq_k -A$.

In [1], we define Paraconsistent Logic Programs by extending the syntax of antitonic logic programs to an arbitrary complete bilattice of truth-values. Such programs are interpreted by *Paraconsistent Well-Founded Models* (WFM^p). In the remainder of this work, we assume that every bilattice is complete, admits conflation and negation, and that these two operators commute. We start by defining an algebraic characterisation of the syntax of paraconsistent logic programs, and resorting to a simplified version of signature.

Definition 2 (PLP signature). *A signature Σ is a set of pairs $\langle f, n \rangle$ such that no function symbol f occurs in two distinct pairs and its arity or rank n is a natural number ($n \geq 0$). The set Σ_n denotes the set of function symbols of arity n. A paraconsistent logic program signature, PLP signature for short, is a signature containing at least the pairs $\langle \neg, 1 \rangle, \langle -, 1 \rangle$ and $\langle \otimes_k, 2 \rangle$.*

So, paraconsistent logic programs will be built using function symbols that represent at least the negation and conflation operators, and the meet under knowledge ordering. The user is free to add other function symbols and constants to the language, defining a specific PLP signature. The association of meaning to the symbols of a PLP signature is captured by the definition of PLP algebra:

Definition 3 (PLP algebra). *Given a PLP signature Σ and a bilattice $\mathfrak{B} = \langle B, \leq_t, \leq_k \rangle$, a PLP algebra $\mathfrak{A}_\Sigma^\mathfrak{B}$ is a pair $\langle B, i \rangle$ where i is an interpretation function assigning functions to functions symbols as follows:*

1. *Each symbol c in Σ_0 (i.e., c is a constant symbol) is interpreted as an element $i(c)$ in B, denoted by $\overset{\bullet}{c}$.*
2. *Each function symbol f in Σ_n, $n \geq 1$, is interpreted as a function $i(f) : B^n \to B$, denoted by $\overset{\bullet}{f}$. Additionally, $\overset{\bullet}{f}$ must be either isotonic or antitonic in each argument with respect to $\leq_k$;*

In particular, $i(\neg)$ and $i(-)$ must be respectively mapped to the negation and conflation operators of $\mathfrak{B}$, and $i(\otimes_k)$ to the meet operator under knowledge ordering of $\mathfrak{B}$.

Note that PLP algebras are related to semantical aspects of the operators. The next step consists in defining the alphabet of symbols of our language, given a PLP signature.

Definition 4 (PLP Alphabet). *Let Σ be a PLP signature and $\Phi = \cup_i \Sigma_i$ ($i \in \mathbb{N}$), the set of function symbols in Σ. Given a set Π of propositional symbols such that $\Pi \cap \Phi = \{\}$, the PLP alphabet $\mathcal{A}_\Sigma^\Pi$ over Π is the disjoint union $\Pi \cup \Phi$.*

The formulae which appear in the body and heads of rules in paraconsistent logic programs are defined inductively as follows:

Definition 5 (PLP Formulae). *Let Σ be a signature and Π a set of propositional symbols. The set of PLP formulae $FORM_{\Sigma}^{\Pi}$ is the least set of strings over the corresponding PLP alphabet $\mathcal{A}_{\Sigma}^{\Pi}$ such that*

- *Every propositional symbol in Π is a formula in $FORM_{\Sigma}^{\Pi}$;*
- *Every constant symbol $c \in \Sigma_0$ is a formula in $FORM_{\Sigma}^{\Pi}$;*
- *If $F_1, \ldots, F_n$ $(n \geq 1)$ are formulae in $FORM_{\Sigma}^{\Pi}$ and f is a n-ary function symbol ($f \in \Sigma_n$), then $f\ F_1\ \ldots\ F_n$ is a formula in $FORM_{\Sigma}^{\Pi}$;*

Given the definitions above, Paraconsistent Logic Programs are defined as follows:

Definition 6 (Paraconsistent Logic Programs). *Let Σ be a PLP signature, $\mathfrak{B} = \langle B, \leq_t, \leq_k \rangle$ a bilattice and $\mathfrak{A}_{\Sigma}^{\mathfrak{B}}$ a PLP algebra. A paraconsistent logic program P over the set of propositional symbols Π is a set of rules of the form $L \leftarrow \Psi$ where L and Ψ are formulas in $FORM_{\Sigma}^{\Pi}$, such that:*

1. *The head L has the form A or $\neg A$, where $A \in \Pi$, i.e. the head of a rule is constituted by a propositional symbol or its negation.*
2. *The body Ψ is an arbitrary formula of $FORM_{\Sigma}^{\Pi}$.*

The present definition of paraconsistent logic programs is simpler than that exhibited in [2]. However, as it was proven there, they have the same expressive power. Throughout this work, we reserve the symbol $\mathfrak{R}$ to denote the PLP algebra below:

Example 1. Let Σ be the following PLP signature:

$$\Sigma = \{\ \langle \neg, 1 \rangle\ ,\ \langle -, 1 \rangle\ \ , \langle \otimes_k, 2 \rangle \} \cup \{ \langle < a, b >, 0 \rangle\ \mid a, b \in [0,1] \}$$

Consider the bilattice $\mathfrak{B} = \left\langle [0,1]^2, \leq_k, \leq_t \right\rangle$ constructed from the cartesian product of the unit interval $[0,1]$ with itself, and the usual total ordering among real numbers. The PLP algebra $\mathfrak{R} = \left\langle [0,1]^2, i \right\rangle$ constructed from $\mathfrak{B}$ such that

1. $i(< a, b >) =< a, b >$.
2. $i(\neg) : [0,1]^2 \rightarrow [0,1]^2$ such that $i(\neg)(< \alpha, \beta >) =< \beta, \alpha >$.
3. $i(-) : [0,1]^2 \rightarrow [0,1]^2$ such that $i(-)(< \alpha, \beta >) =< 1 - \beta, 1 - \alpha >$.
4. $i(\otimes_k) : [0,1]^2 \times [0,1]^2 \rightarrow [0,1]^2$ such that $i(\otimes_k)(< \alpha, \beta >, < \gamma, \delta >) =< \min(\alpha, \gamma), \min(\beta, \delta) >$, where the function $\min : [0,1] \times [0,1] \rightarrow [0,1]$ returns the minimum of its arguments.

Assuming the set $\{$*accept*, *for_reforms*, *against_reforms*, *cut_deficit*$\}$ of propositional symbols, we can now obtain in more formal terms the paraconsistent logic program of the Introduction [4].

In the next definition, we annotate propositional symbols to separate isotonic from antitonic occurrences of them.

[4] For more explanations, the reader is invite to take a look in [2].

Definition 7 (Isotonic and Antitonic occurrences). *Consider a set of propositional symbols Π and let $\Pi^{\pm}$ be the set of annotated propositional symbols $\Pi^{\pm} = \{A^+ \mid A \in \Pi\} \cup \{A^- \mid A \in \Pi\}$. The functions $^{\pm} : FORM_{\Sigma}^{\Pi} \to FORM_{\Sigma}^{\Pi^{\pm}}$ and $^{\mp} : FORM_{\Sigma}^{\Pi} \to FORM_{\Sigma}^{\Pi^{\pm}}$ are defined inductively by:*

- *For a propositional symbol A, $(A)^{\pm} = A^+$ and $(A)^{\mp} = A^-$.*
- *For a constant symbol c, $(c)^{\pm} = c$ and $(c)^{\mp} = c$.*
- *For every function symbol f of arity $(n \geq 1)$ and for all formulae $F_1, \ldots, F_n$ in $FORM_{\Sigma}^{\Pi}$, then*

  $$(f\ F_1\ \ldots\ F_n)^{\pm} = f\ G_1\ \ldots\ G_n \qquad \textit{and} \qquad (f\ F_1\ \ldots\ F_n)^{\mp} = f\ H_1\ \ldots\ H_n$$

  such that for every $1 \leq i \leq n$, formula G_i is $F_i^{\pm}$ and H_i is $F_i^{\mp}$ if $\dot{f}$ is isotonic in the i-th argument, and G_i is $F_i^{\mp}$ and H_i is $F_i^{\pm}$ if $\dot{f}$ is antitonic in the i-th argument.

A propositional symbol A annotated with A^+ (resp. A^-) in $\Psi^{\pm}$ is said to be an isotonic (resp. antitonic) occurrence of A in Ψ.

We use the expression $\Psi[A_1, \ldots, A_m \mid B_1, \ldots, B_n]$ $(m, n \geq 0)$ to denote the formula Ψ, where $A_1 \ldots A_m$ and $B_1, \ldots, B_n$ are, respectively, the isotonic and antitonic occurrences, of propositional symbols in Ψ in the order they appear in Ψ (left to right).

Regarding the semantics, we follow a paraconsistent and paracomplete approach inspired by $WFSX_p$ [3,6], one of the well-founded based semantics proposed for extended logic programs. A technique used in logic programming literature resorts to the notion of partial interpretation, requiring first the usual notion of interpretation.

Definition 8 (Interpretation). *Consider a bilattice $\mathfrak{B} = \langle B, \leq_t, \leq_k \rangle$ and a set of propositional symbols Π. An interpretation of Π is a mapping $I : \Pi \to B$. The set of all interpretations of the propositional symbols with respect to $\mathfrak{B}$ is denoted by $\mathcal{I}_{\mathfrak{B}}$.*

The pointwise ordering extends the knowledge ordering $\leq_k$ on the truth-values in $\mathfrak{B}$ to the set of interpretations as follows.

Definition 9 (Lattice of Interpretations). *Consider $\mathcal{I}_{\mathfrak{B}}$ the set of all interpretations of a set of propositional symbols Π with respect to a bilattice $\mathfrak{B}$, and two interpretations $I_1, I_2 \in \mathcal{I}_{\mathfrak{B}}$. Then, $\langle \mathcal{I}_{\mathfrak{B}}, \sqsubseteq \rangle$ is the complete lattice where $I_1 \sqsubseteq I_2$ iff $\forall_{p \in \Pi}\ I_1(p) \leq_k I_2(p)$. The least interpretation Δ maps every propositional symbol to the least element under $\leq_k$ of $\mathfrak{B}$, and the greatest interpretation ∇ maps every propositional symbol to the top element under $\leq_k$ of the same complete bilattice of truth-values $\mathfrak{B}$.*

A partial interpretation is a pair of interpretations mapping propositional symbols to elements in the underlying bilattice.

Definition 10 (Partial Interpretations). *A partial interpretation I^p of a set of propositional symbols Π is a pair $I^p = \langle I^t, I^{tu} \rangle$ of interpretations of Π with respect to a bilattice $\mathfrak{B}$. The set of all partial interpretations is $\mathcal{I}^p_{\mathfrak{B}}$.*

When speaking about (partial) interpretations we usually omit the set of propositional symbols Π, which is implicitly provided. Sometimes we refer to I^t as the T-component and to the I^{tu} as the TU-component of a partial interpretation. The T-component represents what certainly holds in the partial interpretation, while the TU-component contains what may hold (i.e. the "complement" of what certainly does not hold). It is also important to mark that ($I^t \sqsubseteq I^{tu}$) is not imposed and, consequently, paraconsistency is allowed, i.e. something may certainly hold and not hold. We resort to the following ordering among partial interpretations:

Definition 11. *Let I_1 and I_2 be two partial interpretations. The Fitting's ordering among partial interpretations is defined by: $I_1 \sqsubseteq_f I_2$ iff $I_1^t \sqsubseteq I_2^t$ and $I_2^{tu} \sqsubseteq I_1^{tu}$. The set of partial interpretations ordered by $\sqsubseteq_f$ is a complete lattice. Clearly, the bottom and top elements of this lattice is $\perp_f =< \Delta, \nabla >$ (viz. all is undefined) and $\top_f =< \nabla, \Delta >$ (viz. all is true and false).*

A precise inductive definition of the evaluation of formula in a partial interpretation can now be provided:

Definition 12 (Valuation). *For a given partial interpretation $I^p =< I^t, I^{tu} >$ define inductively the function $val_{I^p} : FORM_{\Sigma}^{\Pi^{\pm}} \to B$ as follows:*

- *For every propositional symbol A, $val_{I^p}(A^+) = I^t(A)$*
- *For every propositional symbol A, $val_{I^p}(A^-) = I^{tu}(A)$*
- *For every constant symbol c, $val_{I^p}(c) = \overset{\bullet}{c}$*
- *For every function symbol f of arity ($n \geq 1$) and for all formulae $F_1, \ldots, F_n$ in $FORM_{\Sigma}^{\Pi^{\pm}}$, then $val_{I^p}(f\ F_1\ \ldots\ F_n) = \overset{\bullet}{f}\ (val_{I^p}(F_1), \ldots, val_{I^p}(F_1))$*

The valuation function $\widehat{I^p} : FORM_{\Sigma}^{\Pi} \to B$ is defined by $\widehat{I^p}(\Psi) = val_{I^p}(\Psi^{\pm})$. If the formula Ψ is of the form $\Psi[A_1, \ldots, A_m \mid B_1, \ldots, B_n]$ by the expression $\widehat{\Psi}[I(A_1), \ldots, I(A_m) \mid J(B_1), \ldots, J(B_n)]$ we mean $\widehat{\langle I, J \rangle}(\Psi)$, where $\langle I, J \rangle$ is a partial interpretation constructed from the interpretations I and J.

The main property of the valuation function is captured in the following proposition, explaining why we designated isotonic and antitonic occurrences of propositional symbols those annotated with $^+$ and $^-$, respectively:

Proposition 1.[2] *Let Ψ be a formula in $FORM_{\Sigma}^{\Pi}$, and I^p and J^p partial interpretations such that $I^p \sqsubseteq_f J^p$. Then we conclude that $val_{I^p}(\Psi^{\pm}) \leq_k val_{J^p}(\Psi^{\pm})$ and $val_{I^p}(\Psi^{\mp}) \geq_k val_{J^p}(\Psi^{\mp})$. Therefore, $\widehat{I^p}(\Psi) \leq_k \widehat{J^p}(\Psi)$.*

The notion of model of a program is now straightforward:

Definition 13 (Model). *A partial interpretation I^p satisfies a rule $L \leftarrow \Psi$ of a paraconsistent logic program P iff $\widehat{I^p}(\Psi) \leq_k \widehat{I^p}(L)$. A partial interpretation I^p is a model of P iff I^p satisfies all rules of P.*

Considering the semantical equivalence of the rules $\neg A \leftarrow \Psi$ and $A \leftarrow \neg\Psi$ (i.e. $\widehat{I^p}(\Psi) \leq_k \widehat{I^p}(\neg A)$ iff $\widehat{I^p}(\neg\Psi) \leq_k \widehat{I^p}(A)$), in order to simplify notation, we assume that every rule $\neg A \leftarrow \Psi \in P$ was replaced by $A \leftarrow \neg\Psi$. In other words, in the rest of this work, we admit only rules with propositional symbols in the head.

The rationale below is to extract from the program all its consequences, assuming that an interpretation is used to evaluate the antitonic occurrences of propositional symbols. This is achieved by coupling together the ideas of Gelfond-Lifschitz division operators [13] and of the immediate consequences operator of van Emden and Kowalski [19], along the lines of [12].

Definition 14 (Gamma operator). *Let P be a paraconsistent logic program and J a given interpretation. The generalised immediate consequences operator $T^{\mathfrak{B}}_{\frac{P}{J}} : \mathcal{I}_{\mathfrak{B}} \rightarrow \mathcal{I}_{\mathfrak{B}}$ maps interpretations to interpretations:*

$$T^{\mathfrak{B}}_{\frac{P}{J}}(I)(A) = lub_k \left\{ \left[\widehat{<I, J>}(\Psi) \mid A \leftarrow \Psi \in P \right] \right\}$$

The $\Gamma^{\mathfrak{B}}_P$ operator is defined as the least fixed point (lfp) of $T^{\mathfrak{B}}_{\frac{P}{J}}$ with respect to the $\sqsubseteq$ ordering between interpretations, formally:

$$\Gamma^{\mathfrak{B}}_P(J) = lfp\ T^{\mathfrak{B}}_{\frac{P}{J}} = T^{\mathfrak{B}}_{\frac{P}{J}} \uparrow^{\lambda}, \text{ for some ordinal } \lambda$$

In conformity with [17], negation and default negation are not unrelated: the Coherence Principle, when adopted, ensures that the former entails the latter. From an epistemic viewpoint, coherence can be seen as an instance of the Necessitation Principle, which states that if something is known then it is believed, i.e. its negation is false in all minimal models [4]. Therefore, one has $A \leq_k not\ \neg A$ (Coherence Principle). Since $not\ \neg A =_{def} -A$, it suffices to guarantee $A \leq_k -A$ then. Technically, we say a model $M^p = \langle M^t, M^{tu} \rangle$ of P complies with the Coherence Principle iff for every propositional symbol A appearing in the language of P, $\widehat{M^p}(A) \leq_k \widehat{M^p}(-A)$.

Given these arguments, and in order to enforce coherence, we will resort to the semi-normal gamma operator, inspired by the approach taken in [3, 6]:

Definition 15 (Semi-normal Gamma operator). *Let P be a paraconsistent logic program and J an interpretation. The semi-normal immediate consequences operator $T^{\mathfrak{B}}_{\frac{P_s}{J}} : \mathcal{I}_{\mathfrak{B}} \rightarrow \mathcal{I}_{\mathfrak{B}}$ maps interpretations to interpretations:*

$$T^{\mathfrak{B}}_{\frac{P_s}{J}}(I)(A) = lub_k \left\{ \left[\widehat{<I, J>}(\Psi \otimes_k -A) \mid A \leftarrow \Psi \in P \right] \right\}$$

We also define $\Gamma^{\mathfrak{B}}_{P_s}(J) = lfp\ T^{\mathfrak{B}}_{\frac{P_s}{J}} = T^{\mathfrak{B}}_{\frac{P_s}{J}} \uparrow^{\lambda}$, for some ordinal λ

Mark that coherence is enforced in every propositional symbol A by not letting the value of the bodies for A surpass the value of $-A$. From the monotonicity of the $\Gamma_P^{\mathfrak{B}}\Gamma_{P_s}^{\mathfrak{B}}$ operators [2] we know, again by the Knaster-Tarski theorem [18], for any paraconsistent logic program P, that $\Gamma_P^{\mathfrak{B}}\Gamma_{P_s}^{\mathfrak{B}}$ has a least fixed point. This fixed point is affirmed to define the paraconsistent well-founded semantics:

Definition 16 (Paraconsistent Well-founded Semantics). *Consider a paraconsistent logic program P and a partial interpretation $M^p = \langle M^t, M^{tu}\rangle$. We say M^p is a partial paraconsistent stable model of P iff $M^t = \Gamma_P^{\mathfrak{B}}(\Gamma_{P_s}^{\mathfrak{B}}(M^t))$ and $M^{tu} = \Gamma_{P_s}^{\mathfrak{B}}(M^t)$. The least partial paraconsistent stable model under the Fitting ordering is the paraconsistent well-founded model $WFM^p(P)$, and can be obtained by iterating $\Gamma_P^{\mathfrak{B}}\Gamma_{P_s}^{\mathfrak{B}}$ from Δ. Then, given that interpretation I_w is the least fixed point of $\Gamma_P^{\mathfrak{B}}\Gamma_{P_s}^{\mathfrak{B}}$ under $\sqsubseteq$ ordering, we shall have $WFM^p(P) = \langle I_w, \Gamma_{P_s}^{\mathfrak{B}}(I_w)\rangle$. If $M^p = \langle M^t, M^{tu}\rangle$ is a partial paraconsistent stable model, we say a propositional symbol A is strictly overdefined wrt. M iff $M^{tu}(A) <_k M^t(A)$.*

The rationale is the following. The "computation" is started assuming that all propositional symbols are assigned the least element under the knowledge ordering. By a first application of the $\Gamma_{P_s}^{\mathfrak{B}}$ operator, an upper approximation to the values of propositional symbols is determined in order to satisfy the rules in the program. This upper approximation is then used in the $\Gamma_P^{\mathfrak{B}}$ step to increase the lower assignment to the propositional symbols. This is the net effect of the $\Gamma_P^{\mathfrak{B}}\Gamma_{P_s}^{\mathfrak{B}}$ combination. This process is iteratively continued by reducing the upper assignment to the propositional symbols via $\Gamma_{P_s}^{\mathfrak{B}}$ while at same time enforcing coherence, and then using this interpretation to get a better lower approximation in resorting to $\Gamma_P^{\mathfrak{B}}$. This process is transfinitely applied till the least fixed point is reached. This idea of approximation is fully explored in [10].

Example 2. Let P be the paraconsistent logic program presented in Example 1. According to Definition 16, we have

$M^t(accept) =< 0.87, 0.85 >, M^{tu}(accept) =< 0.15, 0.13 >$
$M^t(cut\_deficit) =< 0.87, 0 >, M^{tu}(cut\_deficit) =< 0.15, 0 >$

where both *accept* and *cut_deficit* are strictly overdefined. In the next section, such values will be characterised as inconsistent or dependent on inconsistency.

3 Capturing inconsistency and its dependencies

Following the motivation set forth in the introduction, next we prove results intended to furnish satisfactory answers to the questions arisen there. Whilst proving them, the bilattice of truth-values is assumed *infinitary k-distributive*[5]. We also suppose that an inconsistent constant γ is only allowed to appear in rules of the form $A \leftarrow \gamma$. The purpose is to keep a tougher control on the propagation,

[5] A bilattice $\mathfrak{B} = \langle B, \leq_t, \leq_k\rangle$ will be *infinitary k-distributive* if the lattice $\langle B, \leq_k\rangle$ is infinitary distributive.

via the program rules, of inconsistency generated by constants. Acting this way, we force inconsistent constants to behave as ordinary inconsistent propositional symbols. We recall that we are admitting every rule $\neg A \leftarrow \Psi \in P$ was replaced by $A \leftarrow \neg\Psi$.

We make use of Theorem 1 to ensure that, given a paraconsistent well-founded model $M^p = \langle M^t, M^{tu}\rangle$ for a program P, $M^t \sqsubseteq M^{tu}$ is equivalent to saying every propositional symbol in the language of P is consistent.

Theorem 1. *Let $M^p = \langle M^t, M^{tu}\rangle$ be the least fixed point of $\Gamma_P^{\mathfrak{P}}\Gamma_{P_s}^{\mathfrak{P}}$ for a paraconsistent logic program P. Then $M^t \sqsubseteq \Gamma_s M^t$ iff for every propositional symbol A in P language, $M^t(A) \leq_k \dot{-}\, M^t(A)$.*

Obviously, if $M^t(A) \not\leq_k M^{tu}(A)$ there is some propositional symbol B inconsistent in M^p. In the next proposition we find out a bit more: whenever $M^t(A) \not\leq_k M^{tu}(A)$, either A is itself inconsistent or A depends on some inconsistency. However, before introducing it, notice that not all rules for A effectively contribute to the final value assigned to it according to $WFM^p = \langle M^t, M^{tu}\rangle$. Hence, without affecting the final result, we can eliminate these rules for A.

In the proposition below, we just define the set U_A in order to gather all rules for A contributing to the value established for A relative to WFM^p. The objective is to guarantee that if A is consistent, and for all rules r which effectively contribute to A ($r \in U_A$), if we have for all propositional symbols A' in the body of r, $M^t(A') \leq_k M^{tu}(A')$, then $M^t(A) \leq_k M^{tu}(A)$.

Proposition 2. *Let P be a paraconsistent logic program and A an propositional symbol of its language. Given $M^p = \langle M^t, M^{tu}\rangle$, the WFM^p of P, we define $Body_A = \{\widehat{M^p}(\Psi) \mid A \leftarrow \Psi \in P\}$, and $U_A = \{A \leftarrow \Psi \mid \widehat{M^p}(\Psi)$ is upper bound of $Body_A\}$. If both of the following conditions hold:*

- *$M^t(A) \leq_k \dot{-}\, M^t(A)$*
- *For every rule $A \leftarrow \Psi[B_1, \ldots, B_m | C_1, \ldots, C_n]$ in U_A, we have $M^t(B_i) \leq_k M^{tu}(B_i)$ and $M^t(C_j) \leq_k M^{tu}(C_j)$*

then $M^t(A) \leq_k M^{tu}(A)$.

In other words, if $M^t(A) \not\leq_k M^{tu}(A)$, then A is either inconsistent or there is some rule r for A such that for at least a propositional symbol A' in the body of r, $M^t(A') \not\leq_k M^{tu}(A')$. Recalling Example 2, we have $M^{tu}(accept) =< 0.15, 0.13 ><_k M^t(accept) =< 0.87, 0.85 >$, and $M^{tu}(cut\_deficit) =< 0.15, 0 > <_k M^t(cut\_deficit) =< 0.87, 0 >$. From Proposition 2, we conclude that *accept* is inconsistent and that *cut_deficit* depends on inconsistency (in this case, *accept*). Consequently, any decision that a rational agent takes as regards *cut_deficit*, he should consider that although *cut_deficit* is consistent in M^p, it depends on some inconsistency.

Because of the complexity of the bilattice structure, we do not have the reverse result for Proposition 2 [2]. However we can still have the desired result in specific situations, when, for instance, we assume for every propositional symbol

"A" in the program's language, $M^{tu}(A) \leq_k M^t(A)$, i.e., A is overdefined or $M^t(A) = M^{tu}(A)$.

Before providing this result, notice that it is not enough to have a rule $r = A \leftarrow \Psi[A_1, \ldots, A_m | B_1 \ldots, B_n]$ in a Paraconsistent Logic Program to say A depends on A_i (B_j) according to $WFM^p = M^p$ of P. A reason is that this rule may not effectively contribute to the value established for A in M^p, i.e., $r \notin U_A$, where U_A is the set of effective rules for A (see Proposition 2 for more details). Furthermore, even if a rule $r \in U_A$, we may have situations in which the values assigned to some A_is (B_js) are "absorbed" by other values assigned to other propositional symbols present in the body of r as in the example below:

Example 3. Considering the PLP algebra $\mathfrak{R}$, let P be the following paraconsistent logic program:

$$P = \begin{cases} a \leftarrow < 0.6, 0.6 > \qquad b \leftarrow < 0.3, 0.3 > \\ c \leftarrow \otimes_k \; a \; b \end{cases}$$

In the rule $c \leftarrow a \otimes_k b$, we have $M^t(c) = M^{tu}(c) =< 0.3, 0.3 >$, where $M^t(a) =< 0.6, 0.6 >$ (resp. $M^{tu}(a) =< 0.4, 0.4 >$) is completely absorbed by $M^t(b) =< 0.3, 0.3 >$ (resp. $M^{tu}(b) =< 0.3, 0.3 >$) in this rule. We cannot say c depends on a, because a does not contribute to the value assigned to c in WFM^p for P. So we need a device to capture the exact notion of dependence.

Definition 17. *Let P be a paraconsistent logic program, $M^p = \langle M^t, M^{tu} \rangle$ its paraconsistent well-founded model, and r a rule for A in P. Consider also the set $Ov(r)$ of propositional symbols strictly overdefined in M^p found in r. Taking $M^t(A) = lub_k\{\widehat{\Psi}[M^t(A_1), \ldots M^t(A_m) \mid M^{tu}(B_1), \ldots, M^{tu}(B_n)] \mid A \leftarrow \Psi \in P\}$, we say $Ov(r)$ is effective for A in M^p iff after replacing $M^t(A_i)$ (resp. $M^{tu}(B_j)$) by $M^{tu}(A_i)$ (resp. $M^t(B_j)$) for every $A_i \in Ov(r)$ (resp. $B_j \in Ov(r)$) in $\widehat{\Psi}[M^t(A_1), \ldots M^t(A_m) \mid M^{tu}(B_1), \ldots, M^{tu}(B_n)]$), results into a value distinct of $M^t(A)$.*

The interval $[M^{tu}(C), M^t(C)]$ can be seen as that of the reasonable values that C can assume in WFM^p. Hence if, for all $C \in Ov(r)$ for a rule $A \leftarrow \Psi$, we replace $M^t(C)$ by $M^{tu}(C)$, and even so, no perturbation is observed in $M^t(A)$, then any reasonable value for C does not effectively contribute to conclude A. In Example 3, if one substitutes $< 0.6, 0.6 >$ for $M^t(a) =< 0.4, 0.4 >$, $M^t(c)$ and $M^{tu}(c)$ will continue unaltered. Consequently, c does not actually depend on the inconsistent value assigned to a, for the value of c could be exactly the same if we replace $M^t(a)$ by the consistent value $< 0.4, 0.4 >$. We say A depends on inconsistency in M^p if there exists a rule r for A whose $Ov(r)$ is effective.

Taking this information into account we present the next theorem, allowing to detect whether or not a propositional symbol depends on contradiction.

Theorem 2. *Let P be a paraconsistent logic program with $WFM^p = \langle M^t, M^{tu} \rangle$ where for every propositional symbol B in P[6], $M^{tu}(B) \leq_k M^t(B)$. If $M^t(A) = M^{tu}(A)$, then for every rule $r = A \leftarrow \Psi$ in P, we have*

$$M^t(A) \leq_k \overset{\bullet}{-}\, M^t(A) \text{ and } Ov(r) \text{ is non-effective to } A \text{ in } M.$$

Thus, if A is inconsistent in M^p ($M^t(A) \not\leq_k \overset{\bullet}{-}\, M^t(A)$) or A depends on inconsistency (there exists a rule r for A whose $Ov(r)$ is effective), then $M^{tu}(A) <_k M^t(A)$ since it is assumed that for every propositional symbol B in P, $M^{tu}(B) \leq_k M^t(B)$. Returning to program P of Example 2, obviously we have the set $\{accept\}$ is effective for $cut\_deficit$ in M^p, resulting in $M^{tu}(cut\_deficit) =< 0.15, 0 > <_k M^t(cut\_deficit) = < 0.87, 0 >$.

From the results exhibited in this section, we guarantee the existence of a powerful mechanism to detect inconsistency and its dependencies for a Paraconsistent Logic Program. With a glimpse at its WFM^p, one can know which propositional symbols are inconsistent and which depend on inconsistency. As the reader can check, if we did not abide by the *Coherence Principle* by employing $\Gamma_P^{\mathfrak{P}} \Gamma_P^{\mathfrak{P}}$ to characterise WFM^p, instead of resorting to the semi-normal Gamma operator ($\Gamma_{P_s}^{\mathfrak{P}}$), we would not meet the conditions for keeping track of the propagation of inconsistency.

4 Conclusions and further work

In their various characterisations, inconsistencies augur that something may be running amok. Through a simple example of Paraconsistent Logic Program, interpreted by its WFM^p [1, 2], we motivated how dangerous it can be to draw conclusions when neglecting their dependence on inconsistency. In this paper, by checking for some results, we show how our semantics allows for paraconsistency at the same time it keeps, to some extent, a monitoring eye on inconsistent information and its propagation by program rules. To wit, in a paraconsistent well-founded model, when the TU-component for a certain atom is less, in the knowledge order sense, than its T-component, then the atom's value is inconsistent, or depends on inconsistent atoms. In fact, we can view these results as a generalisation, to our framework for Paraconsistent Logic Programs, of the results set forth in [6].

By warning the rational agent about the existence of inconsistency, and of information depending on inconsistency, WFM^p allows it to take a decision with much more safety. It can also be a fundamental tool in systems based on belief revision, which we intend to explore in a future work.

Acknowledgments. João Alcântara is a PhD student at UNL supported by CAPES - Brasília, Brazil. The authors thank the TARDE and FLUX projects, sponsored by FCT/MCES, Portugal.

[6] In fact, it is enough to consider only the propositional symbols present in the dependence graph of A (including A itself).

References

1. J. Alcântara, C. V. Damásio, and L. M. Pereira. Paraconsistent logic programs. In S. Flesca, S. Greco, N. Leone, and G. Ianni, editors, *8th European Conference on Logics in Artificial Intelligence*, pages 345–356. Springer-Verlag, LNAI 2424, 2002.
2. J. Alcântara, C. V. Damásio, and L. M. Pereira. An encompassing framework for paraconsistent logic programs. (Submitted).
3. J. J. Alferes, C. V. Damásio, and L. M. Pereira. A logic programming system for non-monotonic reasoning. *Special Issue of the Journal of Automated Reasoning*, 14(1):93–147, 1995.
4. J. J. Alferes, L. M. Pereira, and T. C. Przymusinski. 'Classical' negation in non-monotonic reasoning and logic programming. *Journal of Automated Reasoning*, 20:107–142, 1998.
5. W. Carnielli and J. Marcos. A taxonomy of C-systems. In W. Carnielli, M. Coniglio, and I. D'Ottaviano, editors, *Paraconsistency: The logical way to the inconsistent. Proceedings of the II World Congress on Paraconsistency (WCP2000)*, pages 1–94. Marcel Dekker, 2002. http://www.cle.unicamp.br/e-prints/abstract_5.htm.
6. C. V. Damásio. *Paraconsistent Extended Logic Programming with Constraints*. PhD thesis, Universidade Nova de Lisboa, 1996.
7. C. V. Damásio and L. M. Pereira. A survey of paraconsistent semantics for logic programs. In D. Gabbay and P. Smets, editors, *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, volume 2, pages 241–320. Kluwer Academic Publishers, 1998.
8. C. V. Damásio and L. M. Pereira. Antitonic logic programs. In T. Eiter and M. Truszczynski, editors, *Procs. of the 6th Int. Conf. on Logic Programming and Nonmonotonic Reasoning*, pages 748–759. Springer, LNCS/LNAI, 2001.
9. Diário de Notícias' homepage. http://dn.sapo.pt/homepage/homepage.asp?codEdicao=504.
10. M. Denecker, V. Marek, and M. Truszczyński. Approximations, stable operators, well-founded fixpoints and applications in nonmonotonic reasoning. In J. Minker, editor, *Logic-Based Artifical Intelligence*, pages 127–144. Kluwer Academic Publishers, 2000.
11. M. Fitting. Bilattices and the semantics of logic programming. *The Journal of Logic Programming*, 11:91–116, 1991.
12. M. Fitting. The family of stable models. *Journal of Logic Programming*, 17:197–225, 1993.
13. M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R. Kowalski and K. A. Bowen, editors, *5th International Conference on Logic Programming*, pages 1070–1080. MIT Press, 1988.
14. M. Ginsberg. Multivalued logics: A uniform approach to reasoning in artificial intelligence. *Computational Intelligence*, 4:265–316, 1988.
15. CGTP's homepage. http://www.cgtp.pt/GG021210/resultados1.htm.
16. L. Lakshmanan and F. Sadri. On a theory of probabilistic deductive databases. *Theory and Practice of Logic Programming*, 1(1):5–42, 2001.
17. L. M. Pereira and J. J. Alferes. Well-founded semantics for logic programs with explicit negation. In B. Neumann, editor, *European Conference on Artificial Intelligence*, pages 102–106, Wien, Austria, 1992. John Wiley & Sons.
18. A. Tarski. Lattice-theoretic fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.
19. M. van Emden and R. Kowalski. The semantics of predicate logic as a programming language. *Journal of ACM*, 4(23):733–742, 1976.