On 3-valued paraconsistent Logic Programming

Marcelo E. Coniglio
Kleidson E. Oliveira

Institute of Philosophy and Human Sciences and Centre For Logic, Epistemology and the History of Science, UNICAMP, Brazil

Support: FAPESP

Syntax Meets Semantics 2016
Summary

1. **Introduction**
2. **Presenting MPT0 and QMPT0**
3. **A Resolution Calculus for QMPT0**
4. **Interpretation and Models**
5. **Declarative Semantics**
6. **The Work Goes On**
7. **References**
Application of Non-classical Logics in logic programs is more delicate that appears.

Several proposals can be seem in Kifer and Subrahmanian [9], Ginsber [8] and Fitting [7]. 3-valued logic programming was considered in Przymusinski [10] and Delahaye and Thibau [6]. Paraconsistent logic programming was also investigated by Blair and Subrahmanian [1] and by Damásio and Pereira [5].
Based on previous studies of Rodrigues [11] on the foundations of Paraconsistent Logic Programming based on several paraconsistent logics in the so called hierarchy of Logics of Formal Inconsistency (LFIs, see Carnielli, Coniglio and Marcos [3]), we will describe in this talk some results on the theory of clausal resolution and on Logic Programming system based on paraconsistent logic QMPT0.
At first, we will present the two negations to be considered, $\neg$ and $\sim$. Being that $\neg$ is the weak negation and $\sim$ is the strong negation. The tables are as follows:
<table>
<thead>
<tr>
<th></th>
<th>¬P</th>
<th>¬¬P</th>
<th>¬¬¬P</th>
<th>¬¬¬¬P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>0</td>
<td>1</td>
<td>B</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

The third value "B" (both) is distinguished, thus the weak negation of B is B and the strong negation of B is 0. We assume that B is an intermediate value between 0 and 1.
The 3-valued matrix logic MPT0 (see [4]) can be defined over the signature \( \{\land, \lor, \rightarrow, \neg, \sim\} \) in the domain \( \{1, B, 0\} \) in which \( D = \{1, B\} \) is the set of distinguished values. The tables that interpret the connectives are as follows:
### Connectives of MPT0

<table>
<thead>
<tr>
<th>$\rightarrow$</th>
<th>1</th>
<th>B</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>B</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>B</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\land$</th>
<th>1</th>
<th>B</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>B</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>B</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\lor$</th>
<th>1</th>
<th>B</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>B</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$p$</th>
<th>$\neg p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$p$</th>
<th>$\sim p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
The equivalence connective $\equiv$ is defined as

$$(\alpha \equiv \beta) =_{\text{def}} (\alpha \to \beta) \land (\beta \to \alpha),$$

whose table is as follows:

<table>
<thead>
<tr>
<th>$\equiv$</th>
<th>1</th>
<th>$B$</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>$B$</td>
<td>0</td>
</tr>
<tr>
<td>$B$</td>
<td>$B$</td>
<td>$B$</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Now, before we give the axiomatization to MPT0, we have to remember the axiomatization of \( \text{mbC} \).
The basic (LFIs) is the (propositional) logic \( \text{mbC} \), defined over the signature \( \{\land, \lor, \to, \neg, \circ\} \) as follows:

**Axioms:**

(A1) \( \alpha \to (\beta \to \alpha) \)

(A2) \( (\alpha \to \beta) \to ((\alpha \to (\beta \to \gamma)) \to (\alpha \to \gamma)) \)

(A3) \( \alpha \to (\beta \to (\alpha \land \beta)) \)

(A4) \( (\alpha \land \beta) \to \alpha \)

(A5) \( (\alpha \land \beta) \to \beta \)

(A6) \( \alpha \to (\alpha \lor \beta) \)

(A7) \( \beta \to (\alpha \lor \beta) \)

(A8) \( (\alpha \to \gamma) \to ((\beta \to \gamma) \to ((\alpha \lor \beta) \to \gamma)) \)

(A9) \( \alpha \lor (\alpha \to \beta) \)
Axioms: (Cont.)

(A10) \( \alpha \lor \neg \alpha \)

(bc1) \( \circ \alpha \rightarrow (\alpha \rightarrow (\neg \alpha \rightarrow \beta)) \)

Inference Rule:

\[
\text{(MP)} \quad \frac{\alpha, \alpha \rightarrow \beta}{\beta}
\]

Observe that (A1)-(A9) plus (MP) is positive classical logic
Now, a sound and complete Hilbert calculus for MPT0, called LPT0, will be defined.

**Definition - The calculus LPT0 for MPT0**

Let $\Sigma_1$ be the signature \{\(\land\), \(\lor\), \(\rightarrow\), \(\neg\), \(\sim\)\}. The Hilbert calculus LPT0 over $\Sigma_1$ is defined by taking axiom schemas (A1)-(A10) from mbC, MP, plus the following:

(These axiom schemas are present in the book Paraconsistent Logic: Consistency, contradiction and negation, by Carnielli and Coniglio [2])
Axiom schemas:

\[
\begin{align*}
\alpha \lor \neg\alpha & \quad \text{(TND)} \\
\alpha \rightarrow (\neg\alpha \rightarrow \beta) & \quad \text{(exp)} \\
\neg\neg\alpha & \rightarrow \alpha \quad \text{(dneg)} \\
\neg\neg\alpha & \rightarrow \alpha \quad \text{(cf)} \\
\alpha & \rightarrow \neg\neg\alpha \quad \text{(ce)} \\
\neg(\alpha \lor \beta) & \rightarrow (\neg\alpha \land \neg\beta) \quad \text{(neg}_1 \lor) \\
(\neg\alpha \land \neg\beta) & \rightarrow \neg(\alpha \lor \beta) \quad \text{(neg}_2 \lor) \\
\neg(\alpha \land \beta) & \rightarrow (\neg\alpha \lor \neg\beta) \quad \text{(neg}_1 \land) \\
(\neg\alpha \lor \neg\beta) & \rightarrow \neg(\alpha \land \beta) \quad \text{(neg}_2 \land) \\
\neg(\alpha \rightarrow \beta) & \rightarrow (\alpha \land \neg\beta) \quad \text{(Ir} \rightarrow) \\
(\alpha \land \neg\beta) & \rightarrow \neg(\alpha \rightarrow \beta) \quad \text{(Ip} \rightarrow)
\end{align*}
\]
From this we have the following:

**Lemma**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>¬¬A ≡ A</td>
</tr>
<tr>
<td>2</td>
<td>∼∼A ≡ A</td>
</tr>
<tr>
<td>3</td>
<td>¬∼A ≡ A</td>
</tr>
</tbody>
</table>
From this we have the following:

**Lemma**

1. $\neg\neg A \equiv A$
2. $\sim\sim A \equiv A$
3. $\neg\sim A \equiv A$
4. $(A \land B) \lor C \equiv (A \lor C) \land (B \lor C)$
5. $(A \lor B) \land C \equiv (A \land C) \lor (B \land C)$
6. $\neg(A \lor B) \equiv \neg(A) \land \neg(B)$
7. $\neg(A \land B) \equiv \neg(A) \lor \neg(B)$
8. $\sim(A \lor B) \equiv \sim(A) \land \sim(B)$
9. $\sim(A \land B) \equiv \sim(A) \lor \sim(B)$
In MPT0 we can introduce the following notions:

- A *literal* is a formula of the form $A, \neg A, \sim A$ or $\sim \neg A$, in which $A$ is a atomic formula. In each case it is said that the literal contains the atomic formula $A$.

- Literals of the form $A$ or $\neg A$ are called *positive*, the others are called *negative*.

- A formula $A$ is called *atom* when there are only positive literals in $A$.

This is motivated by the fact that, in MPT0, there are only four formulas (up to equivalence) based on $p$ constructed with $\neg$ and $\sim$: $p$, $\neg p$, $\sim p$ and $\sim \neg p$. As you can see again in the table of negations.
Presenting MPT0 and QMPT0

### Table of Negations

<table>
<thead>
<tr>
<th>$P$</th>
<th>$¬P$</th>
<th>$\sim P$</th>
<th>$\sim¬P$</th>
<th>$\sim\sim P$</th>
<th>$¬¬P$</th>
<th>$¬\sim P$</th>
<th>$¬¬¬P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>0</td>
<td>1</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Presenting MPT0 and QMPT0

A Resolution Calculus for QMPT0

Interpretation and Models

Declarative Semantics

The Work Goes On

References

- A clause of MPT0 is a formula of the form:

\[ L_1 \lor \cdots \lor L_k \lor \sim L_{k+1} \lor \cdots \lor \sim L_{k+m} \]

such that each \( L_i \) is a positive literal in MPT0.

- A clause is called positive (negative) if it contains only positive (negative) literals.

- A set \( S \) of clauses is called satisfiable if there is a valuation on MPT0 such that \( v(K) \in \{1, B\} \) for all clauses \( K \) in \( S \). In that case \( v \) is called a model of \( S \).

- A clause \( K \) in MPT0 is consequence of a set of clauses \( S \) (denoted by \( S \models_{MPT0} K \)), if for all valuations \( v \), if \( v(S) \subseteq D \) then also holds \( v(K) \in D \).
The MPT0 extension to first-order logic is known as QMPT0. The semantics of QMPT0 (which will be described very briefly here) is given by the so-called *pragmatic structures*, introduced by da Costa, Mikenberg and Chuaqui in the context of the theory of quasi-truth. Afterwards it was generalized by Coniglio and Silvestrini.
A pragmatic structure is an structure $\mathcal{A} = \langle D, (\cdot)^{\mathcal{A}} \rangle$ appropriate to interpret the first order languages in Tarskian style, in which $D$ is a non-empty set (the domain of $\mathcal{A}$) and $(\cdot)^{\mathcal{A}}$ interprets the symbols of the language in the usual way, with this difference: each $n$-ary predicate $p$ in the language is interpreted as a triple $p^{\mathcal{A}} = \langle p^+_A, p^-_A, p_B^A \rangle$ where $p_+$, $p_-$ and $p_B$ are mutually disjoint sets such that $p_+ \cup p_- \cup p_B = D^n$. The elements of $p_+$, $p_-$ and $p_B$ are the $n$-uples that satisfy $p$, do not satisfy $p$, and that simultaneously satisfy and do not satisfy $p$, respectively.
Appropriate operations are defined between such triples for interpreting the connectives and the quantifiers. Thus, a formula $\varphi$ with free variables $x_1, \ldots, x_n$ generates a triple $\varphi^A = \langle \varphi^+_A, \varphi^-_A, \varphi^B_A \rangle$ such that
Presenting MPT0 and QMPT0

A Resolution Calculus for QMPT0

Interpretation and Models

Declarative Semantics

The Work Goes On

References

- $\varphi^+_A = \{ \bar{a} \in D^n : A \models \varphi[\bar{a}] \text{ and } A \not\models \neg \varphi[\bar{a}] \}$;
- $\varphi^-_A = \{ \bar{a} \in D^n : A \not\models \varphi[\bar{a}] \text{ and } A \models \neg \varphi[\bar{a}] \}$;
- $\varphi^+_B = \{ \bar{a} \in D^n : A \models \varphi[\bar{a}] \text{ and } A \models \neg \varphi[\bar{a}] \}$;
- $\varphi^+_A \cup \varphi^+_B = \{ \bar{a} \in D^n : A \models \varphi[\bar{a}] \}$;
- $\varphi^-_A \cup \varphi^-_B = \{ \bar{a} \in D^n : A \models \neg \varphi[\bar{a}] \}$.
A closed (or ground) term is a term with no occurrences of variables. A closed formula is a formula that not contains free occurrences of variables. The universal and existential closure of a formula are defined as usual.
In QMPT0 we introduce the following notions:

- A first-order *literal* is a formula of the form $A$, $\neg A$, $\sim A$ or $\sim \neg A$, in which $A$ is an atomic formula of the first-order language. In each case it is said that the literal contains the atomic formula $A$.
- Literals of the form $A$ or $\neg A$ are called *positive*, the others are called *negative*.
- A positive literal is also called, a *atom*.
- A clause of QMPT0 is a closed formula of the form:

$$\forall x_1 \ldots \forall x_n (L_1 \lor \cdots \lor L_k \lor \neg L_{k+1} \lor \cdots \lor \neg L_{k+m})$$

such that each $L_i$ is a positive literal in QMPT0 and $x_1, \ldots, x_n$ are all the variables occurring in

$$(L_1 \lor \cdots \lor L_k \lor \neg L_{k+1} \lor \cdots \lor \neg L_{k+m}).$$

The usual notation we will use for clauses, equivalent to the above presented is:

$$\forall x_1 \cdots \forall x_n (L_1 \lor \cdots \lor L_k \leftarrow L_{k+1} \land \cdots \land L_{k+m})$$

or just

$$L_1, \ldots, L_k \leftarrow L_{k+1}, \ldots, L_{k+m}$$
- A clause is called *positive* (*negative*) if it contains only positive (negative) literals.
- A set $S$ of clauses is called *satisfiable* if there is a pragmatic structure $\mathcal{A}$ such that $\mathcal{A} \models K$ for all clauses $K$ in $S$. In that case $\mathcal{A}$ is called a *model* of $S$.
- A clause $K$ is a consequence from a set of clauses $S$ (denoted by $S \models_{QMPT0} K$), if for all models $\mathcal{A}$ of $S$ also holds that $\mathcal{A} \models K$. 
A *definite program clause* is a clause of the form

\[ L \leftarrow K_1, \ldots, K_n \]

containing exactly one atom in its consequent. The positive literal \( L \) is called *head* and \( K_1, \ldots, K_n \) is called *body* of the program clause.

Alternatively, a definite program clause may be represented as

\[ (\neg)A \leftarrow (\neg)A_1, \ldots, (\neg)A_n \]

where \((\neg)A\) denotes one of the literals \( A \) (atomic formula) or \( \neg A \) (paraconsistent negation of an atomic formula).
A *unit* is a clause of the form

\[ L \leftarrow \]

that is, a definite program clause with empty body.

A *definite program* \( \mathcal{P} \) is a finite set of definite programs clauses.
The empty clause, denoted by □, is a clause whose antecedent and consequent are empty and it can be interpreted as a classic contradiction in the sense that it is unsatisfiable.

Let $\alpha$ be a first-order formula without quantifiers in a signature with at least one individual constant. We define $S_{\alpha} = \{\bar{\alpha} : \bar{\alpha} \text{ is a ground instance } \alpha\}$. If $\Gamma$ is a set of formulas without quantifiers, then $\bar{\Gamma} = \bigcup\{S_{\alpha} : \alpha \in \Gamma\}$. 
Proposition

Given a set of clauses $S$, $	ilde{S}$ is satisfiable in MPT0 (as a set of propositional clauses) if and only if $S$ is satisfiable in QMPT0.
Inspired by the approach to paracomplete 3-valued clausal resolution introduced in 1986 by P. H. Schmitt in [12], we set a resolution calculus for QMPT0. For this, just a basic resolution rule will be considered as an inference rule, as can be seen in the coming slides. Since the clauses are implicitly universally quantified, an auxiliary concept will be necessary.
Definition

Two literals of QMPT0, $L_1$ and $L_2$, are said to be \textit{complementary} if one of the following conditions holds:

1. $L_1$ is positive and $L_2$ is $\sim L_1$.
2. $L_2$ is positive and $L_1$ is $\sim L_2$. 
Below are two examples of resolution rules:

\[
\begin{array}{ll}
L \lor \bigvee_{i=1}^{n} A & K \lor \bigvee_{j=1}^{m} \neg A \\
& \frac{L \lor K}{L \lor K}
\end{array}
\]

\[
\begin{array}{ll}
L \lor \bigvee_{i=1}^{n} \neg A & K \lor \bigvee_{j=1}^{m} \neg A \\
& \frac{L \lor K}{L \lor K}
\end{array}
\]

Formally, we have:
A Resolution Calculus for QMPT0

Definition

Let $K_1 = L_{1,1} \lor \ldots \lor L_{1,n}$ and $K_2 = L_{2,1} \lor \ldots \lor L_{2,r}$ be two clauses. A clause $K$ is obtained from $K_1$ and $K_2$ through a basic resolution step if there are literals $L_1$ and $L_2$, with $L_i$ occurring in $K_i$ ($i = 1, 2$), and a substitution $\sigma$ such that $\sigma(L_1)$ and $\sigma(L_2)$ are complementary literals, being $\sigma$ the most general unifier with this property. In this case, the resolvent is $K = \sigma(K_0)$, where $K_0$ is the disjunction of literals that appear in $K_1$ (unless all instances of $L_1$ as literal in $K_1$) or in $K_2$ (unless all instances of $L_2$ as literal in $K_2$). We say that $K$ is a basic resolvent of $K_1$ and $K_2$. 
From $K_1$ and $K_2$ we obtain $K$ by a general resolution step if there are renaming substitutions $\mu_1$ and $\mu_2$ such that $K$ can be obtained from $\mu_1(K_1)$ e $\mu_2(K_2)$ by a basic resolution step.
Definition

For a set $S$ of clauses:
- $Res(S)$ denotes the closure of $S$ by resolution
- $Subst(S)$ is the set of all the substitution instances of clauses in $S$.

Lemma

In QMPT0, a set of clauses $S$ is satisfiable iff $Res(S)$ is satisfiable.
The importance of the Paraconsistent third-excluded law

Let \( P_1 \) and \( P_2 \) be the programs:

\[
P_1 = \begin{cases} 
A \leftarrow C \\
A \leftarrow \neg C 
\end{cases} 
\quad \text{and} \quad 
P_2 = \begin{cases} 
A \leftarrow \neg C \\
C \leftarrow A 
\end{cases} 
\]

We need \( C \lor \neg C \) to derive \( A \lor A \) in the first program and \( C \lor C \) in the second, as it can be seen in the following derivations by resolution:
This lead us to define the following:
Definitions

- Given a set of clauses $S$, we define the support for $S$ as the set $\text{Sup}(S) = \{ p(x_1, \ldots, x_n) : p$ is a $n$-ary predicate symbol that occurs in $S \}$. 

- Let $S$ be a set of clauses, we define $S_+$ as the set $S \cup \{ p(x_1, \ldots, x_n) \lor \neg p(x_1, \ldots, x_n) : p(x_1, \ldots, x_n) \in \text{Sup}(S) \}$. In other words $S_+$ denotes the union of $S$ with relevant instances of the third-excluded law.
Thanks to $\text{Sup}(S)$, we guarantee the completeness of resolution.

**Theorem - Completeness of Clausal Resolution in QMPT0, version 1**

Let $S$ be a set of clauses. Then, $S_+$ is satisfiable in QMPT0 iff the empty clause does not belong to $\text{Res}(S_+)$. 
A Resolution Calculus for QMPT0

Theorem - Completeness of Clausal Resolution in QMPT0, version 2

Let $S$ be a set of clauses which is satisfiable in QMPT0, and let $L$ be a ground literal (that is, without variables). Then, $S \models_{QMPT0} L$ iff $\bigvee_{i=1}^{k} L \in Res(S_+)$ for some $k \geq 1$. 

Marcelo E. Coniglio and Kleidson E. Oliveira

On 3-valued paraconsistent Logic Programming
Corollary - Completeness of Clausal Resolution in QMPT0, version 3

Let $S$ be a set of clauses in QMPT0, and let $L$ be a ground literal. Then, $S \models_{QMPT0} L$ iff the empty clause belongs to $Res(S_+ \cup \{\sim L\})$. 

Marcelo E. Coniglio and Kleidson E. Oliveira
On 3-valued paraconsistent Logic Programming
Now we define *Herbrand Universe* and *Herbrand Base* for a first order language $\mathcal{L}$.

**Definition - Herbrand Universe**

The *Herbrand Universe* $U_\mathcal{L}$ for $\mathcal{L}$ is the set of all ground terms that can be formed from constant and function symbols that appear in $\mathcal{L}$.

**Definition - Herbrand Base**

The Herbrand base $B_\mathcal{L}$ for QMPT0 is the set of all ground positive literals that can be formed using the predicate symbols of QMPT0 with the ground terms of Herbrand Universe as parameters.
Consider now the ground atoms \(\{P_1, P_2, P_3, P_4\}\) and the following situation:

\(I = \{P_1, P_2, \neg P_3\}\) is a partial interpretation. (No information about \(P_4\))

\(I = \{P_1, P_2, \neg P_3, P_4\}\) is a total interpretation.

\(I = \{P_1, P_2, \neg P_3, P_4, \neg P_4\}\) is a total interpretation.

This induces the following definitions.
Definition

Given a program $\mathcal{P}$, a subset $\mathcal{I}$ of $B_{\mathcal{P}}$ is called a *Herbrand partial interpretation*. If $\mathcal{I}$ contains all the ground literals that are logical consequences of the program $\mathcal{P}$ in QMPT0, $\mathcal{I}$ is called a *Herbrand partial model*.

Given the Herbrand base $B_{\mathcal{L}}$, we have that $B_{\mathcal{L}}^+$ is a subset of $B_{\mathcal{L}}$ formed by the ground atomic formulas. Then we can define a *Herbrand pragmatic interpretation*. 
Definition

A **Herbrand (pragmatic) interpretation** for $\mathcal{L}$ is a subset $\mathcal{I}$ of $B^+_\mathcal{L}$ with the following property: for each $A \in B^+_\mathcal{L}$, either $A \in \mathcal{I}$ or $\neg A \in \mathcal{I}$. 
Any Herbrand interpretation $\mathcal{I}$ generates in fact a pragmatic structure $\mathfrak{A}$ as follows: for each $n$-ary predicate symbol $p$,

- $p^+_\mathfrak{A} = \{(t_1, \ldots, t_n) \in U^n_L : p(t_1, \ldots, t_n) \in \mathcal{I} \text{ and } \neg p(t_1, \ldots, t_n) \notin \mathcal{I}\}$;
- $p^-_\mathfrak{A} = \{(t_1, \ldots, t_n) \in U^n_L : p(t_1, \ldots, t_n) \notin \mathcal{I} \text{ and } \neg p(t_1, \ldots, t_n) \in \mathcal{I}\}$;
- $p_B^\mathfrak{A} = \{(t_1, \ldots, t_n) \in U^n_L : p(t_1, \ldots, t_n) \in \mathcal{I} \text{ and } \neg p(t_1, \ldots, t_n) \in \mathcal{I}\}$.

Observe that either $p(t_1, \ldots, t_n) \in \mathcal{I}$ or $\neg p(t_1, \ldots, t_n) \in \mathcal{I}$.
With some abuse of language, when we have a definite program $\mathcal{P}$, we will refer to the Herbrand universe $U_\mathcal{P}$ and the Herbrand base $B_\mathcal{P}$ of $\mathcal{P}$.

**Definition**

Let $\mathcal{L}$ be a first-order language and $S$ the set of closed formulas in $\mathcal{L}$. A **Herbrand model** for $S$ is a Herbrand interpretation for $\mathcal{L}$ that is a model of $S$. 
Lemma

Let $S$ be a set of clauses and suppose that $S$ has a model. Then $S$ has a Herbrand model.

Proof:
Let $\mathcal{A}$ be a pragmatic interpretation which is a model of $S$. We define a Herbrand interpretation $\mathcal{I}$ as follows:

$$\mathcal{I} = \{ L \in B_L : \mathcal{A} \models L \}$$

Using that $\mathcal{A}$ is a model of $S$, it follows that the pragmatic structure $\mathcal{A}_\mathcal{I}$ generated by $\mathcal{I}$ is also a model of $S$. 
Each classic logic program is associated with a monotonic function that plays a very important role in the theory. This technique is adapted to the case of QMPT0.
Definition

Let $\mathcal{P}$ be a definite program. The mapping $T_\mathcal{P} : 2^{B_\mathcal{P}} \rightarrow 2^{B_\mathcal{P}}$ is defined as follows.

$T_\mathcal{P}(I) = \{ L \in B_\mathcal{P} : \exists K \in \bar{\mathcal{P}} (\text{head}(K) = L \text{ and } \text{body}(K) \subseteq I) \}$.

Proposition

Let $\mathcal{P}$ be a definite program. Then the mapping $T_\mathcal{P}$ is monotonic and continuous.
Definition

Given a program $\mathcal{P}$, the *Least partial Model for $\mathcal{P}$* is the set

$$M_\mathcal{P} = \{ L \in B_\mathcal{P} : \mathcal{P} \models_{\text{QMPT0}} L \}.$$  

Definition

Given a program $\mathcal{P}$, and $\bar{\mathcal{P}}$ the set of ground clauses obtained from it, we define the *ground support of $\mathcal{P}$* as the set

$$\text{Sup}(\mathcal{P}) = \{ A : A \in B^+_\mathcal{P}, \text{ and there exist } K \in \bar{\mathcal{P}} \text{ such that } A \text{ occurs in } K \text{ or } \neg A \text{ occurs in } K \}.$$  

As before, $\text{Sup}(\mathcal{P})$ serves to collect the potential applications of the excluded middle of the form $A \lor \neg A$. 

Marcelo E. Coniglio and Kleidson E. Oliveira
On 3-valued paraconsistent Logic Programming
Suppose that in a certain program \( \mathcal{P} \) we have two potential applications of the third excluded law, say \( G \) and \( \neg G \) and \( H \) and \( \neg H \). This will generate the 4 extended programs:

\[
\begin{align*}
\mathcal{P} & \cup \{G, H\} \\
\mathcal{P} & \cup \{G, \neg H\} \\
\mathcal{P} & \cup \{\neg G, H\} \\
\mathcal{P} & \cup \{\neg G, \neg H\}
\end{align*}
\]

Formally, we have:
Let $\lambda(\mathcal{P})$ be the cardinal of $\text{Sup}(\mathcal{P})$, and let $\text{Sup}(\mathcal{P}) = \{A_i : i < \lambda(\mathcal{P})\}$ be an enumeration of $\text{Sup}(\mathcal{P})$. We define

$$\mathcal{P}^+ = \mathcal{P} \cup \{A_i \vee \neg A_i : i < \lambda(\mathcal{P})\}.$$ 

Finally, for $\gamma \in 2^{\lambda(\mathcal{P})}$ let

$$L_i^\gamma = \begin{cases} A_i & \text{if } \gamma(i) = 0 \\ \neg A_i & \text{if } \gamma(i) = 1 \end{cases}$$

Let $\mathcal{I}_\gamma^\mathcal{P} = \{L_i^\gamma : i < \lambda(\mathcal{P})\}$.

This produces the extended program $\mathcal{P}_\gamma = \mathcal{P} \cup \mathcal{I}_\gamma^\mathcal{P}$, for each $\gamma \in 2^{\lambda(\mathcal{P})}$.

Observe that $\mathcal{P}_\gamma$ is possible infinite.
From the definitions before, immediately we prove the following:

**Proposition**

Let $\mathcal{P}$ be a definite program and $K$ a clause without variables. So: $K \in \text{Res}(\mathcal{P}_+)$ implies that $K \in \text{Res}(\mathcal{P}^+)$. 

We now need to establish some auxiliary technical results on classical first order logic CL and its relationship with QMPT0. To differentiate resolution systems, we denote by $\text{Res}_{\text{QMPT}0}$ and $\text{Res}_{\text{CL}}$ the clausal resolution operators of the first order logics QMPT0 and CL, respectively.
Recall first a classical result.

Theorem

Let $\mathcal{P}$ be a definite program in CL, and $A$ a ground atom. Then,

$$\mathcal{P} \models_{CL} A \iff A \in T_{\mathcal{P}} \uparrow \omega.$$
From that result from classical logic programming, we obtain its analogue for QMPT0. The formulation is more complicated, because of the potential applications of the excluded middle, which forces to consider all extensions $P_\gamma$ from the original program $P$. We must first state an additional Lemma:

**Lemma**

Let $P$ be a definite program in QMPT0, and let $P'$ be the definite program in CL obtained by applying the following translation $(\cdot)':$

- $\neg p(t_1, \ldots, t_n)$ by $p'(t_1, \ldots, t_n)$ and
- $\neg\neg p(t_1, \ldots, t_n)$ by $\neg p'(t_1, \ldots, t_n)$.

So for all ground positive literal $L$, $L \in T_P \uparrow \omega$ in QMPT0 iff $L' \in T_{P'} \uparrow \omega$ in CL.
Finally, we arrive to the following result

**Theorem - Fixpoint Characterization of Herbrand Least Partial Model for QMPT0**

Let $\mathcal{P}$ be a definite program in QMPT0, and $L$ a ground positive literal. Then,

$$\mathcal{P} \models_{QMPT0} L \iff L \in \bigcap_{\gamma \in 2^\lambda(\mathcal{P})} T_{\mathcal{P}, \gamma} \uparrow \omega.$$ 

Observe that all the extensions of $\mathcal{P}$ must be considered, because of the third-excluded law.
Example:
Let \( \mathcal{P} \) the logic program below.

\[
\mathcal{P} = \begin{cases} 
A \leftarrow H \\
C \leftarrow \neg H \\
D \leftarrow C \\
A \leftarrow D, \neg G \\
A \leftarrow G, \neg E \\
\neg E \leftarrow 
\end{cases}
\]
We find the set of ground literal that are a consequence from it, using the characterization given by the previous theorem. To simplify the exposition, we assume that the program is propositional and finite. We will also consider in $\bar{\text{Sup}}(\mathcal{P})$ only the atoms that are needed to generated $M_{\mathcal{P}}$, to shorten the presentation. Note that the set $M_{\mathcal{P}}$ of ground literal deductible from $\mathcal{P}$ is $\{\neg E, A\}$. However, to deduce $A$, two applications of the excluded middle are required: $H \lor \neg H$ and $G \lor \neg G$. Thus, we define the following:
Sup(\mathcal{P}) = \{H, G\} = \{A_0, A_1\}
\lambda(\mathcal{P}) = 2 = \{0, 1\}
A_0 = H \text{ and } A_1 = G
2^{\lambda(\mathcal{P})} = \{(0, 0), (0, 1), (1, 0), (1, 1)\}
\mathcal{I}_{(0,0)} = \{H, G\}, \mathcal{I}_{(0,1)} = \{H, \neg G\}, \mathcal{I}_{(1,0)} = \{\neg H, G\}, \mathcal{I}_{(1,1)} = \{\neg H, \neg G\}.
\[
T_{P(0,0)} \uparrow 0 = \emptyset \\
T_{P(0,0)} \uparrow 1 = \{\neg E, H, G\} \\
T_{P(0,0)} \uparrow 2 = T_{P(0,0)}(\{\neg E, H, G\}) = \{\neg E, A, H, G\} = T_{P(0,0)} \uparrow \omega.
\]

\[
T_{P(0,1)} \uparrow 0 = \emptyset \\
T_{P(0,1)} \uparrow 1 = \{\neg E, H, \neg G\} \\
T_{P(0,1)} \uparrow 2 = T_{P(0,1)}(\{\neg E, H, \neg G\}) = \{\neg E, A, H, \neg G\} = T_{P(0,1)} \uparrow \omega.
\]
Declarative Semantics

\[ T_{P(1,0)} \uparrow 0 = \emptyset \]
\[ T_{P(1,0)} \uparrow 1 = \{ \neg E, \neg H, G \} \]
\[ T_{P(1,0)} \uparrow 2 = T_{P(1,0)}(\neg E, \neg H, G) = \{ \neg E, \neg H, G, A, C \} \]
\[ T_{P(1,0)} \uparrow 3 = T_{P(1,0)}(\{ \neg E, \neg H, G, A, C \}) = \{ \neg E, \neg H, G, A, C, D \} = T_{P(1,0)} \uparrow \omega. \]

\[ T_{P(1,1)} \uparrow 0 = \emptyset \]
\[ T_{P(1,1)} \uparrow 1 = \{ \neg E, \neg H, \neg G \} \]
\[ T_{P(1,1)} \uparrow 2 = T_{P(1,1)}(\{ \neg E, \neg H, \neg G \}) = \{ \neg E, C, \neg H, \neg G \} \]
\[ T_{P(1,1)} \uparrow 3 = T_{P(1,1)}(\{ \neg E, C, \neg H, \neg G \}) = \{ \neg E, C, D, \neg H, \neg G \} \]
\[ T_{P(1,1)} \uparrow 4 = T_{P(1,1)}(\{ \neg E, C, D, \neg H, \neg G \}) = \{ \neg E, C, D, \neg H, \neg G, A \} = T_{P(1,1)} \uparrow \omega. \]

Finally, \( \bigcap T_{P(1,1)} \uparrow \omega = \{ \neg E, A \} = M_P \)
The next steps include defining the procedural semantics of programs for QMPT0 by a combination of SLD and SLI-resolution techniques. From the results obtained and the given examples, it is clear that one of the key issues is to reduce the size of the support $\text{Sup}(\mathcal{P})$ (or $\bar{\text{Sup}}(\mathcal{P})$) without prejudicing the completeness, in order to obtain a more feasible system in terms of implementation.
Thank You!


M. E. Coniglio and L. H. C. Silvestrini.
An alternative approach for quasi-truth.  

C. V. Damásio and L. M. Pereira.  
A survey of paraconsistent semantics for logic programs.  

J. P. Delahaye and V. Thibau.  
Programming in three-valued logic.  

M. C. Fitting.  
Bilattices and the semantics of logic programming.  

M. Ginsber. 
Multivalued logics: A uniform approach to reasoning in artificial intelligence.

*Computational Intelligence, 4:265–316, 1988.*

M. Kifer and V. S. Subrahmanian.
Theory of generalized annotated logic programming and its applications.

T. C. Przymusinski.
Well-founded semantics coincides with three-valued stable semantics.
*Fundamenta Informaticae, 13(4):445–463, 1990.*

T. G. Rodrigues.
Sobre os fundamentos da programação lógica paraconsistente.

P. H. Schmitt.
Computational aspects of three-valued logic.