Many-valued Intuitionistic Implication and Inference Closure in a Bilattice-based Logic

Zoran Majkić
Dept. of Computer Science, UMIACS, University of Maryland, College Park, MD 20742
zoran@cs.umd.edu
http://www.cs.umd.edu/ ~ zoran/

Abstract—In this paper we present a many-valued logic programming, based on reinterpreted Belnap’s 4-valued bilattice: we introduce the new semantics for a 4-valued implication by relative pseudo-complement, used for intuitionistic logics. This kind of logic programming is particularly useful for data integration with possibly incomplete and inconsistent information. We define an ontological encapsulation of the epistemic many-valued logic programs with negation, based on this bilattice, into 2-valued meta logic programs. Obtained 2-valued logic semantically reflects original epistemic many-valued logic, and can be used in order to define many-valued logic entailment and inference closure for many-valued truth assignments.

I. INTRODUCTION

Generally, three-valued, or partial model semantics has had an extensive development for logic programs, [1], [2]. Przymusinski extended the notion of the stable model to allow 3-valued, or partial, stable models, [3], and showed that every program has at least one partial stable model, and that the well-founded model is the smallest among them, [4]. Once one has made the transition from classical to partial models allowing incomplete information, it is a small step towards allowing models admitting inconsistent information: the idea is that we may define the semantics of logic programs with inconsistent information by the smallest partial stable 4-valued models. Doing so provides a natural framework for the semantic understanding of logic programs that are distributed over several sites, with possible conflicting information coming from different places.

By a logic program we mean a finite set of universally quantified clauses of the form \( \forall(A \leftarrow L_1 \land \ldots \land L_m) \), and a set of constraints \( \forall(A \leftarrow L_1 \land \ldots \land L_m) \), where \( m \geq 0 \).

A is an atom and \( L_i \) are positive or negative literals (see [5]). Following a standard convention, such clauses will be simply written as clauses of the form \( A \leftarrow L_1 \land \ldots \land L_m \).

The alphabet \( \mathcal{L} \) of a program \( P \) consists of all constants, predicate and functional symbols that appear in \( P \), a countably infinite set of variable symbols, connectives \( (\land, \lor, \neg, \leftarrow \) i.e., and, or, and not and logic implication, respectively), and the usual punctuation symbols. We assume that if \( P \) does not contain any constant, then one is added to the alphabet. The language \( \mathcal{L} \) of \( P \) consists of all the well-formed formulae of the so-obtained first order theory. We assume that the Herbrand universe is \( \Gamma_U = \Gamma \cup \Omega \), where \( \Gamma \) is ordinary domain of database constants, and \( \Omega \) is an infinite enumerable set of marked null values, \( \Omega = \{\omega_0, \omega_1, \ldots\} \), and for a given logic program \( P \) composed of a set of predicate and function symbols, \( P_S, F_S \) respectively, we define a set of all terms, \( T_S \), and its subset of ground terms \( T_0 \), then atoms are defined as: \( A_S = \{p(c_1, \ldots, c_n) \mid p \in P_S, n = \text{arity}(p) \text{ and } c_i \in T_S\} \).

The Herbrand base, \( H_P \), is the set of all ground (i.e., variable free) atoms.

In [6], Belnap introduced a logic intended to deal in a useful way with inconsistent or incomplete information. It is the simplest example of a non-trivial bilattice and it illustrates many of the basic ideas concerning them. We denote the four values as \( B = \{t, f, \tau, \perp\} \), where \( t \) is true, \( f \) is false, \( \tau \) is inconsistent (both true and false) or possible, and \( \perp \) is unknown.

As Belnap observed, these values can be given two natural orders: truth order, \( \leq_t \), and knowledge order, \( \leq_k \), such that \( f \leq_t \tau \leq_t t \), \( f \leq_t \perp \leq_t t \), and \( \perp \leq_k f \leq_k \tau \), \( \perp \leq_k t \leq_k \tau \).

This two orderings define corresponding equivalences \( =_t \) and \( =_k \). Thus any two members \( \alpha, \beta \) in a bilattice are equal, \( \alpha = \beta \), if and only if (shortly “iff”) \( \alpha =_t \beta \) and \( \alpha =_k \beta \).

Meet and join operators under \( \leq_t \) are denoted \( \land \) and \( \lor \); they are natural generalizations of the usual conjunction and disjunction notions. Meet and join under \( \leq_k \) are denoted \( \otimes \) (consensus, because it produces the most information that two truth values can agree on) and \( \oplus \) (gullibility, it accepts anything it’s told), such that:

\[ f \otimes t = \perp, f \oplus t = \tau, \tau \land \perp = f \text{ and } \tau \lor \perp = t. \]

There is a natural notion of truth negation, denoted \( \sim \), (reverses the \( \leq_t \) ordering, while preserving the \( \leq_k \) ordering): switching \( f \) and \( t \), leaving \( \land \) and \( \lor \), and corresponding knowledge negation, denoted \( \neg \), (reverses the \( \leq_k \) ordering, while preserving the \( \leq_t \) ordering): switching \( f \) and \( t \), leaving \( \land \) and \( \lor \).

These two kind of negation commute: \( \sim x = \sim x \) for every member \( x \) of a bilattice.

It turns out that the operations \( \land, \lor \) and \( \sim \), restricted to \( B^4_t = \{f, t, \perp\} \) are exactly those of Kleene’s strong 3-valued logic. A more general information about bilattice may be found in [7]. The Belnap’s 4-valued bilattice is infinitary distributive. In the rest of this paper we denote by \( B_4 \) (or simply \( B \)) this 4-valued Belnap’s bilattice, and by \( B^4_t, B^4_k \) its 3-valued sublattices (the first is used for a 3-valued Kleene’s logic (programs with incomplete, unknown information), the second for complete but inconsistent information).

As we can see, this bilattice logic does not contain the definition of the many-valued logic implication operator. In this paper we will enrich this bilattice with the implication
operator based on the intuitionistic semantics. A (ordinary) Herbrand interpretation is a many-valued mapping \( I : H_P \rightarrow B \). If \( P \) is a many-valued logic program with the Herbrand base \( H_P \), then the ordering relations and operations in a bilattice \( B_3 \) are propagated to the function space \( B_3^{H_P} \), that is the set of all Herbrand interpretations (functions), \( I = I_B : H_P \rightarrow B_3 \).

**Definition 1** Ordering relations are defined on the Function space \( B_3^{H_P} \) pointwise, as follows: for any two Herbrand interpretations \( v_B, w_B \in B_3^{H_P} \),

1. \( v_B \leq w_B \) if \( v_B(A) \leq w_B(A) \) for all \( A \in H_P \).
2. \( v_B \leq_k w_B \) if \( v_B(A) \leq_k w_B(A) \) for all \( A \in H_P \).
3. \( \sim v_B \), such that \( (\sim v_B)(A) = (v_B(A)) \).
4. \( -v_B \), such that \( (-v_B)(A) = -(v_B(A)) \).

It is straightforward [8] that this makes a function space \( B_3^{H_P} \) itself a complete infinitary distributive bilattice. Ginsberg [9] defined a world-based bilattices, considering a collection of worlds \( W \), where by world we mean some possible way of how things might be:

**Definition 2** [9] A pair \( [U, V] \in \mathcal{P}(W) \times \mathcal{P}(W) \) of subsets of \( W \) (here \( \mathcal{P}(W) \) denotes the powerset of the set \( W \)) express truth of some sentence \( p \), with \( \leq_{\leq} \leq_k \) and truth and knowledge preorders relatively as follows:

1. \( U \) is a set of worlds where \( p \) is true, \( V \) is a set of worlds where \( p \) is false, \( P = U \cup V \) is a set where \( p \) is inconsistent (both true and false), and \( W - (U \cup V) \) where \( p \) is unknown.
2. \( [U, V] \leq [U_1, V_1] \) iff \( U \subseteq U_1 \) and \( V_1 \subseteq V \).
3. \( [U, V] \leq_k [U_1, V_1] \) iff \( U \subseteq U_1 \) and \( V \subseteq V_1 \).

The bilattices operations associated with \( \leq_{\leq} \leq_k \) are:

4. \( [U, V] \land [U_1, V_1] = [U \cup U_1, V \cup V_1] \).
5. \( [U, V] \lor [U_1, V_1] = [U \cup U_1, V \cup V_1] \).
6. \( \sim [U, V] = \neg [V, U] \).

Let denote by \( B_{3^H} \) the set \( \mathcal{P}(W) \times \mathcal{P}(W) \), then the structure \( (B_{3^H}, \land, \lor, \neg, \sim) \) is a bilattice.

Such definition is well suited for 3-valued Kleene logic, but not for 4-valued logic: for example, let \( W = [1, 100] \) be the closed interval of integers (indexes for a collection of worlds), \( U = [1, 60], V = [50, 100], U_1 = [1, 60], V_1 = [40, 100] \), then \( [U, V] \leq [U_1, V_1] \) while \( T - P = [1, 50] \supset [1, 40] = U_1 - P_1 = T_1, F = F_1 = [60, 100] \). The other reason is that in this case we can’t assign \( \tau \) to any sentence, otherwise we will object an inconsistent theory where all sentences are inconsistent.

Consequently, we adopt a triple \( [T, P, F] \) of mutually disjoint subsets of \( W \) to express truth of some sentence \( p \) (the \( W - T \cup P \cup F \) is a set of worlds where \( p \) is unknown), with the following definition for their truth and knowledge orders:

2.1 \( [T, P, F] \leq_{\leq} [T_1, P_1, F_1] \) iff \( T \subseteq T_1 \) and \( F_1 \subseteq F \).
2.2 \( [T, P, F] \leq_{k} [T_1, P_1, F_1] \) iff \( T \subseteq T_1, P \subseteq P_1 \) and \( F \subseteq F_1 \).

The meet and join truth and knowledge operations for this extended bilattice can be found in [10]. In this way we consider the possible value as weak true value and not as inconsistent (that is both true and false), and, consequently, we do not consider them to establish truth ordering (our Def 2.1 differs from Ginsberg’s Def.). We have more knowledge for ground atom with such value, w.r.t. the true ground atom, because we know also that if we assign the true value to such atom we may obtain an inconsistent database.

The difference of a possible and unknown value may be explained also intuitively as follows: if we consider a 3-valued Kleene’s strong logic, and try to use it in order to give a semantics for databases with inconsistencies, then we will obtain a number of stable 3-valued models (minimal ‘repairs’) for it. In each such stable model the set of unknown ground atoms is invariant: if one atom is unknown in some model it remains unknown in all other stable models. But we will have some atom true in some and false in some other stable model: to such atoms we can assign the possible logic value in a framework of this 4-valued logic, in order to obtain a unique minimal Herbrand model. Because of that, I prefer the Lukasiewicz’s term “possible” for bilattice top-knowledge logic value \( \top \), and Kleene’s term “unknown” for bilattice bottom-knowledge logic value \( \bot \).

In [11] is given the definition for modal operators on bilattices, which generalizes both Kripke’s possible world approach and Moore’s autoepistemic logic: a modal operator is any n-ary function from the bilattice \( B \) to itself, with the following property:

**Definition 3** [11] A modal operator on a bilattice \( B \) will be called deductive if and only if it commute with \( \otimes \) and \( \oplus \). All other modal operators will be called nondeductive.

For example, a nondeductive modal operator is Moore’s operator \( \mathcal{M}p \), [12], where \( \mathcal{M}p \) is intended to capture the notion of, “I know that \( p \)”, that is, \( \mathcal{M}p = t \), if value of \( p \) is \( t \) or \( \tau \); otherwise.

This operator we introduced for the Autoepistemic many-valued Logic Programming [13], in order to relax a belief on ground facts and be able to reason in the presence of the inconsistent information also: a ground fact, \( p \leftarrow t \), is substituted by a modal formula \( \mathcal{M}(p \leftarrow t) \). The head of each integrity constrain for such programs that admit inconsistent information is changed from \( f \) to \( \tau \) logic value.

The plan of this paper is the following: Section 2 defines the 4-valued bilattice inference requirements and is defined many-valued intuitionistic implication for rules of a 4-valued logic programming. Such formal definition for many-valued implication is used in Section 3, where is shortly presented model theoretic Herbrand semantics. [14], [15] for the ontologically encapsulated many-valued (EMV) logic programmes (transformation into a 2-valued logic programs). The semantic-reflection of the epistemic into the ontological framework, defines the semantic of this operator \( \mathcal{E} \). Such ontological encapsulation is a necessary tool in order to define the many-valued logic inference, and is used also for coalgebraic semantics (equal to the well-founded semantics) of general logic programs [16].

In Section 4 we discuss the world-based closure for the epistemic truth assignment based on the deduction closure of the 2-valued logic at the ontological level. Finally, in Section 5 is given the answer to the, by Ginsberg and Scharf, open problem: we present the inference closure for the truth assignments in many-valued logics.
II. MANY-VALUED INTUITIONISTIC IMPLICATION

One of the key insights behind bilattices [9], [7] was the interplay between the truth values assigned to sentences and the (non classic) notion of implication. The problem was to study how truth values should be propagated "across" implications. The constructivism is surprisingly close to logic programming [17]. The features of logic programming that are unconventional from the classical point of view find immediate constructivistic explanations. Constructivism does not allow indefiniteness in proofs: from a constructivistic viewpoint implications are not "hidden disjunctions". Constructivism is causalistic: implications are viewed as inferring new information from already proved information, like in logic programming.

In this paragraph we will try to introduce the formal definition of many-valued implication for logic programs. Notice that logic implication and logic entailment, as pointed by Belnap, for the 4-valued logic are strongly connected: the implication has to be the principal structure for inference (entailment) capabilities. Let denote by $\vdash_B$ this bilattice 4-valued entailment, this this paradigm can be defined as follows: " $p \vdash_B q$ if $p \rightarrow q$ is true".

From this point of view, we are fundamentally interested only for the cases when an implication is true. That is, at least from my point of view, the reason why in many-valued logic programming (e.g., Fitting, Przymusinski 3-valued logic) the implication is, differently from other logic connectives, defined as two-valued connective, such that it preserves truth: $p \rightarrow q$ just in case for each assignment of one of four values to the variables, the truth value of $p$ does not exceed the value of $q$ (in symbols: $v(p) \leq v(q)$ for each truth assignment $v : \mathcal{L} \rightarrow B$). In order to obtain such many-valued definition, which generalizes the 2-valued definition given above we will consider the conservative extensions of Łukasiewicz’s and Kleene’s strong 3-valued matrices (where third logic value $\perp$ is considered as unknown) in the intuitionistic way. Such conservative extensions are based on the observation we explained in precedence: the problem to study how the truth values should be propagated "across" implications can be restricted only to true implications (in fact we don’t use implications when they are not true, because the ‘immediate consequence operator’ derives new facts only for true clauses, i.e. when implication is true). Thus, what we must guarantee is that if $b \leftarrow a$ and $c \leftarrow b$ are true then $c \leftarrow a$ also must be true, in order to guarantee the reflexivity and the transitivity of the logic entailment $\vdash_B$.

In other words, that law is intimately connected with inference fixpoint semantics for logic programs: let us consider that in the $i$-th step the ground clause $b \leftarrow a$ become true , so that we derive the new fact $b$ from body $a$, and that in some $i+k$-th step the ground clause $c \leftarrow b$ becomes true, so that we derive the new fact $c$. it means that $c \leftarrow a$ has to be true.

With such constructivistic considerations, Heyting produced an axiomatic system of propositional logic which claimed to generate as theorems precisely those sentences that are valid according to the intuitionistic conception of truth. It is well known that the implication for intuitionistic logic satisfies the conditions above. It is defined by the relative pseudo-complement [18] as follows:

The logical value of intuitionistic implication $a \rightarrow b$ is the greatest member $c$ of $B$ (w.r.t. the truth ordering) such that $a \wedge c \leq b$ (that is $a \rightarrow b \equiv \bigvee \{ c \mid a \wedge c \leq b \}$). Thus we obtain that $a \rightarrow b$ iff $a \leq b$ (for $c = t$), and that the modus ponens inference rule holds: if $a \rightarrow b$ , that is, $a \leq b$ , and $a$ is true, then $t \leq b$ , that is, $b$ must be true.

The relative pseudo-complement for finite distributive lattices always exist. In fact, the intuitionistic implications, $\sim^+_3 \rightarrow^+_3$, for two sublattices $B^+_3$, $B^+_3$, respectively, are uniquely defined by the following truth-functional tables

$$
\begin{array}{|c|c|c|}
\hline
\rightarrow & \top & \bot \\
\hline
\top & \top & \top \\
\bot & \bot & \top \\
\Hline
\end{array}
$$

$$
\begin{array}{|c|c|c|}
\hline
\sim^+ & \top & \bot \\
\hline
\top & \top & \top \\
\bot & \bot & \bot \\
\Hline
\end{array}
$$

Remark: The negation in the intuitionistic logic is defined by the pseudo-complement, that is, $\neg a$ is equivalent to $a \rightarrow f$, i.e., $\neg a$ is the l.u.b. of $\{\beta \mid a \wedge \beta = f\}$, so that $\neg \neg a$ is different from the epistemic negation " $\sim"$ (it is also different from the knowledge bilattice negation $^*\sim$) and, consequently, we do not need it for the logic programming.

For our purpose we obtained the Łukasiewicz’s extension and a tautology $a \leftarrow a$ for any formula $a$. From intuitionistic semantics for this implication we guarantee the truth of a clause (implication) $p(e) \leftarrow B$, whenever (iff) $v_B(p(e)) \geq t$ $v_B(B)$, as used in fixpoint semantics for ‘immediate consequence operators’. It is easy to verify that the 3-valued intuitionistic implications are sceptical reductions of $\leftarrow :$ the values in excess are replaced by $f$ (false).

We define also the restriction to 2-valued logic implication $\leftarrow_2$, such that its matrix, $f_{\leftarrow_2} : B \times B \rightarrow 2$, where $2 = \{t, f\}$, is defined by: for any $\alpha, \beta \in B$, $f_{\leftarrow_2}(\alpha, \beta) = t$ iff $f_{\leftarrow_2}(\alpha, \beta) = f$, otherwise.

Thus, the valuations can be extended to maps from the set of all ground (variable free) formulas to $B$ in the following way:

<table>
<thead>
<tr>
<th>$\rightarrow$</th>
<th>$\top$</th>
<th>$\bot$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\top$</td>
<td>$\top$</td>
<td>$\bot$</td>
</tr>
<tr>
<td>$\bot$</td>
<td>$\bot$</td>
<td>$\bot$</td>
</tr>
</tbody>
</table>

### Definition 4
Let $P_S$ be the set of all predicate symbols ($P_B \subseteq P_S$ is a subset of built-in predicates), $e$ the special (error) constant, and $I : H_P \rightarrow B$ be a many-valued Herbrand interpretation. A valuation $I$ determines:

1. A Generalized interpretation mapping $\mathcal{I} : P_S \times \bigcup_{i \leq \omega} T_0^i \rightarrow B \cup \{e\}$, such that for any $e = (c_1, \ldots, c_n) \in T_0^n$, $\mathcal{I}(p(e)) \equiv I(p(e))$ iff arity $(p) = n$ : $e$ otherwise.
2. A unique valuation map, also denoted $v_B : \mathcal{L} \rightarrow B$, on the set of all ground formulas $\mathcal{L}$, according to the following conditions: $v_B(\sim X) = \sim v_B(X)$ and
EMV denote this new 2-valued encapsulation of many-valued logic. We distinguish between what the reasoner believes in (at the object level; thus, roughly, the ‘meta’ level) and what is actually true or false in the real world (at the EMV ontological ‘meta’ level), thus, the ‘meta’ level is an (classic) encapsulation of the object level.

III. MANY-VALUED ONTOLOGICAL ENCAPSULATION

The many-valued logic is based on four bilattice values which are epistemic. Sentences are to be marked with some of these bilattice logic values, according as to what the computer has been told; or, with only a slight metaphor, according to what it believes or knows.

Of course these sentences have also Frege’s ontological truth-values (true and false), independently of what the computer has been told: we want that the computer can use also these ontological ‘meta’ knowledge. Let, for example, the computer believes that the sentence p has a value τ (possible); then the ‘meta’ sentence, “I (computer) believe that p has a possible value” is ontologically true. The many-valued encapsulation, defined as follows, is just the way to pass from the epistemic (‘object’) many-valued logic into ontological (‘meta’) 2-valued logic.

Such encapsulation is characterized by having capability for semantic-reflection: intuitively, for each predicate symbol we need some function which reflects its logic semantic over a domain. Let us introduce also the set of functional symbols κp in our logical language in order to obtain an enriched logical language where we can encapsulate the many-valued logic programming. Such set of functional symbols will be derived from the following Bilattice-semantic mapping K:

**Definition 5** A semantic-reflection is a mapping \( K : \mathcal{P}_S \rightarrow (\mathcal{B} \cup \{e\}) \cup \{\tau_0\} \), and we denote shortly \( \kappa_p = K(p) : \mathcal{B} \cup \{\tau_0\} \rightarrow \mathcal{B} \cup \{e\}, \) such that for any \( e = (c_1, \ldots, c_n) \in \mathcal{T}_0^n \), holds: \( \kappa_p(e) = e \) if \( \text{arity}(p) \neq n \). If \( p \) is a built-in predicate, then \( \kappa_p \) is uniquely defined by: for any \( e \in \mathcal{T}_0^n, n = \text{arity}(p) \), holds that \( \kappa_p(e) = t \) if \( p(e) \) is true; \( f \) otherwise.

The many-valued ground atoms of a bilattice-based logical language \( \mathcal{L}_B \) can be transformed in ‘encapsulated’ atoms of a 2-valued logic in the following simple way: the original (many-valued) fact that the ground atom \( p(c_1, \ldots, c_n) \), of the n-ary predicate \( p \), has an epistemic value \( \alpha = \kappa_p(c_1, \ldots, c_n) \) in \( \mathcal{B}_S \), we transform in the new ground atom \( p^\mathcal{A}(x_1, \ldots, x_n, \alpha) \) with meaning "it is true that \( p(c_1, \ldots, c_n) \) has a value \( \alpha \). That is, we replace the original n-ary predicate \( p(x_1, \ldots, x_n) \) with n+1-ary predicate \( p^\mathcal{A}(x_1, \ldots, x_n, \alpha) \), with the added logic-attribute \( \alpha \). It is easy to verify that for any given many-valued valuation \( v_S \), every ground atom \( p^\mathcal{A}(c_1, \ldots, c_n, \alpha) \) is ontologically true (when \( \alpha = v_B(p(c_1, \ldots, c_n)) \)) or false. Let EMV denote this new 2-valued encapsulation of many-valued logic.

We distinguish between what the reasoner believes in (at the object (epistemic many-valued sublanguage) level), and what is actually true or false in the real world (at the EMV ontological ‘meta’ level), thus, roughly, the ‘meta’ level is an (classic) encapsulation of the object level.

**Definition 6** Let \( P \) be an ‘object’ many-valued logic program with the set of predicate symbols \( \mathcal{P}_S \). The translation in the encapsulated syntax version in \( \mathcal{P}_A \) is as follows:

1. For any \( \alpha \in \mathcal{B} \), considered as 0-ary predicate (bilattice constant), \( \mathcal{E}(\alpha) = \alpha^\mathcal{A}(\alpha) \), where \( \alpha^\mathcal{A}(\cdot) \) is an unary bilattice predicate over domain of bilattice values, such that for any \( \alpha, \beta \in \mathcal{B} \), \( \alpha^\mathcal{A}(\beta) = t \) if \( \alpha = \beta; \ f \) otherwise;
2. Each positive literal in \( P \), \( \mathcal{E}(p(x_1, \ldots, x_n)) = p^\mathcal{A}(x_1, \ldots, x_n, \kappa_p(x_1, \ldots, x_n)) \);
3. Each negative literal in \( P \), \( \mathcal{E}(\neg p(x_1, \ldots, x_n)) = p^\mathcal{A}(x_1, \ldots, x_n, \neg \kappa_p(x_1, \ldots, x_n)) \);
4. \( \mathcal{E}(\phi \land \varphi) = \mathcal{E}(\phi) \land \mathcal{E}(\varphi) \); \( \mathcal{E}(\phi \lor \varphi) = \mathcal{E}(\phi) \lor \mathcal{E}(\varphi) \);
5. \( \mathcal{E}(\phi \iff \varphi) = \mathcal{E}(\phi) \iff^\mathcal{A} \mathcal{E}(\varphi) \), where \( \iff^\mathcal{A} \) is a new syntax symbol for the implication at the encapsulated 2-valued ‘meta’ level;
6. For any modal formula on ground facts, \( \mathcal{E}(M(p(c_1, \ldots, c_n) \iff t)) = p^\mathcal{A}(c_1, \ldots, c_n) \lor p^\mathcal{A}(c_1, \ldots, c_n, \tau) \).

This embedding of the many-valued logic program \( P \) into a 2-valued ‘meta’ logic program \( \mathcal{P}_A \) is an ontological embedding: it views formulae of \( P \) as beliefs and interprets negation \( \neg p(x_1, \ldots, x_n) \) in rather restricted sense - as belief in the falsehood of \( p(x_1, \ldots, x_n) \), rather as not believing that \( p(x_1, \ldots, x_n) \) is true (like in an ontological embedding for classical negation).

Like for Moore’s autoepistemic operator, for the encapsulation operator \( \mathcal{E}, \mathcal{E} \phi \) is intended to capture the notion of, “I know that \( \phi \) has a value \( v_B(\phi) ^\mathcal{A} \)”, for a given valuation \( v_B \) of the many-valued logic program.

Let \( \mathcal{L} \) be the set of all ground well-formed formulae defined by this Herbrand base \( H_\mathcal{P} \) and bilattice operations (included many-valued implication \( \iff \), with \( \mathcal{B} \subseteq \mathcal{L} \). We define the set of all well-formed encapsulated formulae by:

\( \mathcal{L}^\mathcal{A} = \text{def} \{ \mathcal{E}(\psi) \mid \psi \in \mathcal{L} \}, \) so that \( H_\mathcal{P} \subseteq \mathcal{L}^\mathcal{A} \), thus we obtain

**Proposition 1** The encapsulation operator \( \mathcal{E} : \mathcal{L} \rightarrow \mathcal{L}^\mathcal{A} \) with carrier set of (positive and negative) literals, and ‘meta’ algebra \( \mathcal{L}^\mathcal{A}, \land^\mathcal{A}, \lor^\mathcal{A}, \iff^\mathcal{A} \), where \( \land^\mathcal{A}, \lor^\mathcal{A} \) are 2-valued reductions of bilattice meet and join, respectively, denoted by \( \land, \lor \) also.

2. The ‘meta’ implication operator \( \iff^\mathcal{A} \) is not 2-valued operator: its matrix is given by \( f_{\iff^\mathcal{A}} : \mathcal{B} \times \mathcal{B} \rightarrow 2, \) where \( f_{\iff^\mathcal{A}} = \mathcal{E} \circ f_{\iff} = f_{\iff} \).

Proof: The points 1 and 2 are direct consequences of the Definition 6. The matrix of ‘meta’ implication is equal to the matrix of \( \iff \) (2-valued restriction of \( \iff \)) analogously to 2-valued implication used in Fitting’s fixpoint semantics of immediate consequence operator for logic programs. Moreover, the semantics of ‘meta’ implication is a direct consequence of modal operator \( \mathcal{E} \) and many-valued intuitionistic implication \( \iff \).

Notice, that with the transformation of the original ‘object’ logic program \( P \) into its ontological ‘meta’ version program
A Herbrand interpretation of $P^A$ is a 2-valued mapping (a-interpretation) $I^A : H^B_P \rightarrow \mathbb{2}$. We denote by $2^{H^B_P}$ the set of all interpretations (functions) from $H^B_P$ into $\mathbb{2}$, and by $B^{H^B_P}$ the set of all consistent Herbrand many-valued interpretations, from $H_P$ to the bilattice $B$.

The meaning of the encapsulation of this many-valued logic program $P$ into this ‘meta’ logic program $P^A$ is fixed into the kind of interpretation to give to such new introduced functional symbols $\kappa_p = K(p)$: in fact we want [14] that they reflect (encapsulate) the semantics of the ‘object’ level logic program $P$.

Definition 7 (Satisfaction) The encapsulation of an epistemic ‘object’ logic program $P$ into a ‘meta’ program $P^A$ means that, for any consistent many-valued Herbrand interpretation $I \in B^{H^B_P}$ and its extension $v_B : L \rightarrow B$, the function symbols $\kappa_p = K(p)$, $p \in P_S$ reflect this semantics, i.e. for any tuple $\mathbf{c} \in T_0^{\text{str}(p)}(\mathbf{c})$, $\kappa_0(\mathbf{c}) = I(p(\mathbf{c}))$.

So, we obtain a mapping, $\Theta : B^{H^B_P} \rightarrow 2^{H^B_P}$, such that $I^A = \Theta(I) \in 2^{H^B_P}$, with: for any ground atom $p(\mathbf{c})$, $I^A(E(p(\mathbf{c}))) = t$, if $\kappa_p(\mathbf{c}) = I(p(\mathbf{c}));$ if otherwise.

Let $g$ be a variable assignment which assigns values from $T_0$ to object variables. We extend it to atoms with variables, so that $g(E(p(x_1, ..., x_n))) = E(p(g(x_1), ..., g(x_n)))$, and all formulas in the usual way: $\psi / g$ denotes a ground formula obtained from $\psi$ by assign $g$, then

1. $I^A \vDash g E(p(x_1, ..., x_n)) \iff \kappa_p(g(x_1), ..., g(x_n)) = I(p(g(x_1), ..., g(x_n)))$.
2. $I^A \vDash g E(\neg p(x_1, ..., x_n)) \iff \sim \kappa_p(g(x_1), ..., g(x_n)) = I(p(g(x_1), ..., g(x_n)))$.
3. $I^A \vDash g E(p(x_1, ..., x_n) \land p(x_1, ..., x_n)) \iff \kappa_p(g(x_1), ..., g(x_n)) = I(p(g(x_1), ..., g(x_n)))$.

Notice that in this semantics the ‘meta’ implication $\lnot^A$, in $E(\phi) \iff E(\psi)$, is based on the intuitionistic many-valued implication $\lnot$ which is not classical, i.e., $\phi \not\iff \phi \lor \sim \psi$ determines how the logical value of a body of clause “propagates” to its head.

Remark: The encapsulated logic program, obtained by the transformation of a many-valued logic program, is not the programming language for building logic programs.

It is very different from the Annotated Logic Programming (ALP) [19]: the truth of the encapsulated ground atom $p(\mathbf{c}, \alpha)$, obtained from some atom $p(\mathbf{c})$ of the original many-valued logic program, is relative to the particular many-valued interpretation $I : H_P \rightarrow B$, that is, it is true iff $I(p(\mathbf{c})) = \alpha$, while in ALP, the annotated atom $p(\mathbf{c}, \alpha)$ is true for any interpretation for which $I(p(\mathbf{c})) \leq \alpha$.

Theorem 1 The semantics of encapsulation $E$ is obtained by identifying the semantic-reflection with the $\lambda$-abstraction of Generalized Herbrand interpretation, $K = \lambda I$, so that the semantics of many-valued logic programs can be determined by $I$ (at ‘object’ level) or, equivalently, by its reflection $K$ (at encapsulated or ‘meta’ level).

Proof: From $K = \lambda I$ we obtain that for any $p(\mathbf{c}) \in H_P$ holds $I(p(\mathbf{c})) = \lambda c \in H^B_P(p(\mathbf{c})) = K(p(\mathbf{c}) \leq \kappa_p(c)$, what is the semantics of encapsulation.

We can consider the $\lambda$-abstraction of Generalized Herbrand interpretation as an epistemic semantics, because, given a Herbrand (epistemic) interpretation $I : H_P \rightarrow B$, then for any predicate symbol $p$ and constant $c \in T_0^{\text{str}(p)}$, holds $I(p(\mathbf{c})) = I(p(\mathbf{c}))$. Then the semantic of encapsulation may be defined as follows: “ontological semantic-reflection $\equiv$ epistemic semantics”, that is, $K = \lambda I$

Notice that at ‘meta’ (ontological) level (differently from $\land, \lor$, which are classic 2-value boolean operators), the semantics for ‘meta’ implication operator, $I^A \vDash g E(\phi \iff \psi)$, is not defined on $I^A \vDash g E(\phi)$ and $I^A \vDash g E(\psi)$. For example, let $I^A \vDash g E(p(\mathbf{c}))$ and $I^A \vDash g E(q(\mathbf{d}))$, with $\kappa_p(\mathbf{c}) = f$ and $\kappa_q(\mathbf{d}) = t$; then $p(\mathbf{c}) \iff q(\mathbf{d})$ is false and, consequently, does not hold $I^A \vDash E(p(\mathbf{c}) \iff q(\mathbf{d}))$.

Proposition 2 $I^A \vDash g E(\phi) \iff E(\psi)$ implies $I^A \vDash g E(\phi)$ and $I^A \vDash g E(\psi)$, but not viceversa. The truth of $E(\phi / g)$ and $E(\psi / g)$ are necessary but not sufficient conditions for the truth of $E(\phi / g) \iff E(\psi / g)$.

Following the standard definitions, we say that an interpretation $I^A$, of a program $P^A$, is a model of $P^A$ if and only if every clause of $P^A$ is satisfied in $I^A$. In this way we define a model theoretic semantics for encapsulated logic programs.

A set of formulas $S$, of encapsulated logic EMV, logically entails a formula $\phi$, denoted $S \vDash \phi$, if and only if every model of $S$ is also a model of $\phi$.

IV. MANY-VALUED INFERENCE CLOSURE

Let us consider the bilattice structure $(B_W, \land, \lor, \cdot, +, \sim, \cdot)$ of the Definition 2, where $B_W$ denotes the set $P(W) \times P(W)$, and a collection of world corresponds to the set of Herbrand interpretations over a given Herbrand base $H_P$, i.e., $W = B^{H^B_P}$. Now, with this new setting, we review the general framework proposed by Ginsberg [11].

Let us consider a world-bilattice $B_W$ a set of truth values for formulae in $L^A$. Recall that a truth assignment for the bilattice $B_W$, in this case, is a mapping: $\Phi : L^A \rightarrow B_W$ so that for any $E(\psi) \in L^A$, where $\psi \in L$, the $(U, V) = \Phi(E(\psi))$ is a pair of subsets of $W$: the subset $U$ is the set of worlds (Herbrand interpretations) where $E(\psi)$ is true, while the subset $V$ is the set of worlds (Herbrand interpretations) where $E(\psi)$ is false.

We can project this mapping onto the first component of $B_W = P(W) \times P(W)$ to set another truth assignment $\Phi_+ : L^A \rightarrow P(W)$.

$\Phi_+$ takes a sentence $E(\psi) \in L^A$ and maps it to the collection of worlds where it is known to hold. As a function from $L^A$ to $P(W)$, $\Phi_+$ is nothing more than a relation on $L^A$ and $W$, with $\Phi_+(E(\psi), w)$ if and only if $E(\psi)$ is given as true in the world $w$.

We can easily interpret this as a function $\Phi_+ : W \rightarrow P(L^A)$, which, given a world $w$ (i.e. a Herbrand interpretation),
produces the set of 'meta' (encapsulated) sentences known to be true in it.

Now we are ready to define the world-closed (w-closed) many-valued truth assignment (valuation) \( v_B : \mathcal{L} \rightarrow \mathcal{B} \) given in the Definition 4.

**Definition 8** Let \( v_B : \mathcal{L} \rightarrow \mathcal{B} \) be a truth assignment, obtained as extended valuation of a Herbrand interpretation \( I = w : H_P \rightarrow \mathcal{B} \) (Def. 5), then we define the 'meta' valuation \( v_B^A : \mathcal{L}^A \rightarrow \mathcal{B} \) as follows: for any \( p(c) \in H_P \) and \( \psi, \varphi \in \mathcal{L} \),

1. \( v_B^A(\mathcal{E}(p(c))) = \mathcal{E}(v_B(p(c))) = \kappa_p(c) \);
2. \( v_B^A(\mathcal{E}(\sim p(c))) = \mathcal{E}(v_B(p(c))) = \sim \kappa_p(c) \);
3. \( v_B^A(\mathcal{E}(\psi) \circ \mathcal{E}(\varphi)) = v_B^A(\mathcal{E}(\psi)) \circ v_B^A(\mathcal{E}(\varphi)) \);
4. \( v_B^A(\mathcal{E}(\psi)) = v_B^A(\mathcal{E}(\varphi)) \).

where \( \circ \in \{ \wedge, \vee \} \).

Let \( \mathcal{L}_W = \text{def} \{ \phi = \mathcal{E}(\psi) | \psi \in \mathcal{L} \text{ and } v_B^A(\phi) = t \} \), then a many-valued Herbrand interpretation \( I : H_P \rightarrow \mathcal{B} \) is w-closed if:

(a) \( \Phi^+_w(w) \) is deductively closed (as subset of \( \mathcal{L}^A \)) for each \( w \in W \), and
(b) \( I = \bigcap_{\mathcal{E} \in \mathcal{L}_W} \Phi^+_w(\phi) \), with \( \phi \notin \mathcal{L}_W \) implies \( I \notin \Phi^+_w(\phi) \).

The extended many-valued valuation (truth assignment) \( v_B : \mathcal{L} \rightarrow \mathcal{B} \) of the w-closed Herbrand interpretation \( I : H_P \rightarrow \mathcal{B} \) is w-closed also.

The approach we are following for this 'meta' (encapsulated) 2-valued logic is quite close to that used to define the closure operation in first-order (2-valued) logic itself: the fixed set \( \Phi^+_w(w) \) is deductively closed (that \( \phi \land \sigma \) be in the set if \( \phi \) and \( \sigma \) are, for example) if it is the intersection of all of its closed supersets. This definition makes sense because the intersection of closed sets is itself closed. For many-valued truth assignment closure holds this analog fact:

**Proposition 3** Let \( v_B, u_B \in \mathcal{B}^L \) be two w-closed many-valued truth assignments. Then \( v_B \otimes u_B \) is also w-closed. In general, if \( v_{B_k} \) are all w-closed, then so is \( \prod_k v_{B_k} = v_{B_1} \otimes \cdots \otimes v_{B_n} \),

where, if we represent truth assignments by triples \([T, P, F]\), (of true, possible, and false subsets of \( \mathcal{L} \) respectively; the remaining part of \( \mathcal{L} \) is considered as unknown), then the meet knowledge operation \( \otimes \) is defined by

\[ [T, P, F] \otimes [T_1, P_1, F_1] = [T \cap T_1, P \cap P_1, F \cap F_1] \]

So we can mimic the usual (set-based) definition of closure:

**Definition 9** The w-closure of many-valued truth assignment \( v_B \) is given by

\[ cl(v_B) = \prod \{ u_B | u_B \geq_k v_B \text{ and } u_B \text{ is w-closed} \} \]

In his paper [11] Ginsberg tried to define inference-based closure operation \( cl \) for truth assignments \( v_B : \mathcal{L} \rightarrow \mathcal{B} \), based on the following four principles:

1. It should be a bilattice construction, depending only on the partial orders and operation of bilattice. Here we will consider only Belnap’s 4-valued distributive bilattice.
2. It should accurately reflect the notion of inference already formalized in existing work on extensions to first-order logic.
3. \( cl(cl(v_B)) = cl(v_B) \) for all truth assignments \( v_B \).
4. \( cl(v_B)(\psi) \geq_k v_B(\psi) \) for all \( \psi \in \mathcal{L} \).

where the last two principles are general requirements for any closure operation.

It is easy to verify that one of the conditions on a truth assignment \( v_B \) being w-closed is that \( \Phi^+_w \) be deductively closed. This notion, however, is not bilattice-theoretic. Thus in [11] Ginsberg tried to give an equivalent definition of closure that does not suffer from this deficiency, by the following definition of the apparent closure (a-closure):

**Definition 10** [11] A truth assignment \( v_B : \mathcal{L} \rightarrow \mathcal{B} \) will be called a-closed if:

1. If \( \psi \models_B \phi \), then \( v_B(\phi) \geq_t v_B(\psi) \).
2. \( v_B(\land \psi, \varphi) \geq_k \land \psi, \varphi \).
3. \( v_B(\sim \psi) = \sim v_B(\psi) \).

But, as critically pointed Scherf in [20], these conditions (particularly first one) show very strong assumptions about proposed logic. What we are really imposing here is a classical (2-valued) semantics to a many-valued setting: this framework collapses in the presence of contradictions. If we have a proposition which the top knowledge value \( \top \), then all the propositions will assume the truth value \( \top \). As consequence, the 4-valued Belnap’s logic we can use only three logic values. "For this reason, the search for alternative foundation for Ginsberg’s framework is still wide open and, in our opinion, crucial for future development” concluded Scherf in [20].

What follows is just the positive answer to this open problem, and is valid for any (not only 4-valued Belnap’s logic) many-valued logic.

The notion of logic entailment (or model-theoretic consequence relation) and of logic validity in a 2-valued logic, given by the syntax \( p \models q \). ("q is a consequence of \( p \)"), holds when the set of all models where \( p \) is true is contained in the set of all models where \( q \) is true. The difficulty for many-valued logics is in the understanding of how to define logic entailment when, for instance, the falsity of \( q \) is a consequence of the possibility of \( p \). More over, let \( \Delta \) be the many-valued set of formulae in which each formula has some fixed value (not obligatorily the true value) in \( \mathcal{B} \), so that we can conclude that \( q \) has a possible value: such conclusion is valid if the set of all models where the formulae in \( \Delta \) has these fixed values, is contained in the set of models where \( q \) has a possible value. Such, more complex relation, can be given by the classic 2-valued logic entailment, not for original ('object') many-valued formulae, but for their encapsulated 'meta' formulae, i.e., by the classic 2-valued consequence relation \( \mathcal{E}(\Delta) \models \mathcal{E}(q) \). Thus we are able to substitute the 2-valued (at 'object' level) consequence relation used by Ginsberg in order to define a-closure, with this 'encapsulated' 2-valued consequence relation:

**Definition 11** A truth assignment \( v_B : \mathcal{L} \rightarrow \mathcal{B} \) will be called a-closed if:

\[ if \ E(\psi) \models E(\phi) , then \ v_B^A(E(\psi)) \geq_t v_B^A(E(\phi)). \]

where \( \psi, \phi \in \mathcal{L} \) and \( v_B : \mathcal{L}^A \rightarrow \mathcal{B} \) is the 'meta' valuation determined from \( v_B \), and, \( \psi \models_B \phi \) iff \( E(\psi) \models E(\phi) \).
Notice that the second condition in Ginsberg’s Definition 10 is replaced by the more general (see Belnap’s and Fitting’s considerations) condition 3 of Definition 4; while the third condition is common for all valuations (condition 2 of Definition 4).

**Example for** \( \equiv B \): Let \( \Delta = \{ p_1(c_1), \ldots, p_n(c_n) \} \) be a set of ground atoms, with values \( \alpha_1, \ldots, \alpha_n \) respectively, of a many-valued logic program, so that \( E(\Delta) = E(p_1(c_1)) \wedge \ldots \wedge E(p_n(c_n)) = p_1^A(c_1, \alpha_1) \wedge \ldots \wedge p_n^A(c_n, \alpha_n) \), and \( E(p(c)) = p^A(c, \alpha) \), then \( E(\Delta) \models E(p(c)) \), that is, \( \Delta \models B \ p(c) \), means that the ground atom \( p(c) \) with this value \( \alpha \) is (many-valued) inferred from the set of ground atoms \( \Delta \), if the set of all many-valued models where \( p(c) \) has a value \( \alpha \in B \) contains the set of many-valued models where for each \( 1 \leq i \leq n \), \( p_i(c_i) \) has the value \( \alpha_i \).

Thus, we are able to define the many-valued, bilattice-theoretic dependent, inference closure:

**Theorem 2** A many-valued truth assignment is apparently closed if and only if it is \( w \)-closed. Thus, \( cl(w) = \prod u_B \ u_B \geq w \ v_B \) and \( u_B \) is \( a \)-closed \]. \( cl \) is monotonic, that is, if \( \phi \leq_k \varphi \), then \( cl(\phi) \leq cl(\varphi) \).

Proof: 1. From \( w \)-closed to \( a \)-closed: Let \( v_B : L \rightarrow B \) be \( w \)-closed. If \( E(\psi) \models E(\phi) \), then \( E(\phi) \) will be true in at least as many worlds as \( E(\psi) \) is, and can be false in no more. Thus \( v_B^4(E(\phi)) \leq v_B^4(E(\psi)) \), i.e., \( v_B : L \rightarrow B \) is \( a \)-closed.

2. From \( a \)-closed to \( w \)-closed: Suppose that \( v_B : L \rightarrow B \) is \( a \)-closed, but not \( w \)-closed. Thus, must be some \( w \in W \) such that \( \Phi_+(w) \) is not deductively closed. There must therefore be some collection \( E(\psi_1), \ldots, E(\psi_n) \) with \( E(\psi_i) \in \Phi_+(w) \) and \( E(\psi_1), \ldots, E(\psi_n) \models E(\phi) \) but \( E(\phi) \notin \Phi_+(w) \). But since \( v_B \) is \( a \)-closed, \( \wedge v_B^4(E(\psi_i)) = v_B^4(\wedge E(\psi_i)) = v_B^4(\wedge E(\psi_i)) \leq v_B^4(E(\phi)) \). It follows that from \( E(\psi_i) \in \Phi_+(w) \) \( w \in \bigcap E(\psi) \subseteq \Phi_+(\wedge E(\psi)) = \Phi_+(\wedge E(\phi)) \) or \( w \in \Phi_+(E(\phi)) \), so that \( E(\phi) \) \( \in \Phi_+(w) \) after all. This contradiction completes the proof. The monotonicity of \( cl \) follows immediately from the fact that the set over which the product is taken varies \( k \)-monotonically with the truth assignment in question.

**V. Conclusion**

We have presented a many-valued programming logic with the intuitionistic 4-valued implication, capable of handling incomplete and inconsistent beliefs, based on the revisited 4-valued Belnap’s bilattice, by introducing the possible logic value instead of inconsistent (both true and false), which has clear model theory and fixed point semantics. In the process of the encapsulation we distinguish two levels: the epistemic many-valued level of ordinary logic programs with negation based on a bilattice operators, and the ontological ‘meta’ level of encapsulated logic programs. Such encapsulation is a semantic-reflection at ontological level of the epistemic semantics at many-valued object level, and is a homomorphism between these two logic levels. In this way we are able to formalize the logical entailment for a many-valued logic, by considering the main criticism to Ginsberg’s approach given by Schaefer, consisting in the lack of a multivalued semantics and the use of the notion of validity in 2-valued case even when dealing with more complex sets of truth values. The resulting ontological 2-valued logic, which encapsulates the semantics of many-valued logics is not an issue "per se": it is considered as a minimal logic tool (differently from more complex logics, as annotated and signed logics) useful to define many-valued entailment and inference. But, recently, it is used also in order to define the coalgebraic semantics for logic programs [16], and in the future, considering the fact that for any predicate at many-valued level, by encapsulation we obtain a new predicate, with one logic-attribute extension (for the epistemic logic value), we will consider the possibility to use the ordinary relational database schemas with such extended predicates in global schemas of data integration systems, for query answering from their many-valued Herbrand model.

**References**