A Sequent Calculus for Bilattice-Based Logic and its Many-Sorted Representation

Ekaterina Komendantskaya

Department of Mathematics,
University College Cork,
Ireland
e.komendantskaya@mars.ucc.ie [⋆]

Abstract. We introduce a sequent calculus for bilattice-based annotated logic (BAL). We show that this logic can be syntactically and semantically translated into a fragment MSL* of conventional many-sorted logic MSL. We show deductive equivalence of sequent calculus for BAL and sequent calculus for MSL*.

1 Introduction

Lattice and bilattice-based logics were originally introduced by several independent authors, such as Kifer, Lozinskii, Subrahmanian [15, 16], Fitting [7–9]; Lu, Murray and Rosenthal [20–22] and were further developed in [13, 24, 25]. See also [14, 1] for a very good survey of many-valued proof procedures. These logics were seen as a formalism suitable for automated reasoning about uncertainty. Most of the mentioned approaches made use of the so-called annotated, or signed, languages, where elements of a set of truth values are allowed in the syntax of the language. We will follow the approach of [15, 16, 13, 25] and attach annotations to first-order formulae, such that a typical annotated atom of the language will be of the form $A : \mu$, where A is a first-order atom, and μ represents some annotation taken from a set of truth-values. Complex formulae can be built using these atoms.
We propose a sequent calculus for a first-order bilattice-based annotated logic (BAL). Although the papers we mentioned above provided us with many interesting insights and techniques, the calculus we define here is the first sequent calculus for a full fragment of bilattice-based annotated logics we know of.
Syntactically, we follow [16] and allow not only constant annotations, as in signed logics ([20, 21, 13, 24, 25]), but also annotation variables and terms. We develop the annotated syntax of [16] and enrich it by new bilattice connectives and quantifiers and also we allow annotations over complex first-order formulae, which is a reasonable option for annotated logics.

[⋆] I am grateful to the anonymous referees for their useful comments and suggestions. I thank the grant "Categorical Semantics for Natural Methods of Computation" by the Royal Society/Royal Irish Academy.

Semantically, we work with arbitrary distributive bilattices defined in [10, 11]. We believe that a one-lattice fragment of BAL can be described, with some syntactical restrictions, in terms of lattice-based annotated logics [13, 15, 16, 24, 25]. However, the second lattice constituting the underlying bilattice of BAL determines some novel and non-trivial properties of its models and sequent calculus, see Proposition 2.
The many-valued sequent calculus for BAL was not our primary goal, though. In the second part of the paper we use it for a more general purpose of receiving a conventional syntactical and semantical representation of annotated many-valued logics. Annotated logics are being criticised for allowing semantics interfere with their syntax. Because of their complicated syntax, these logics have not yet received a sufficiently general structural or categorical characterisation, see also [18] for some discussion of this.
Here we continue the work of Manzano [23] translating different non-classical logics into conventional many-sorted logic. We show here that BAL can be semantically, syntactically, and deductively translated into many-sorted logic of [4, 23]. The method of translation can be applied, with minor modifications, to any many-valued annotated logic we mentioned above. Moreover, our results, together with well-known results of Manzano [23] on translation of dynamic (modal) logic and second-order logic into many-sorted logic, include many-valued logics into the family of non-classical logics representable in many-sorted logic. This enables us to compare some properties of (bi)lattice-based logics and the non-classical logics just mentioned.
The structure of the paper is as follows. In Section 2, we define bilattices following [11, 10]. In Section 3, we describe Bilattice-Based Annotated Logic (BAL) and sequent calculus for it. In Section 4, we define a fragment MSL* of the many-sorted logic MSL, which we use for translation of BAL. In the same section, we show the syntactical, semantical and deductive equivalence of BAL and MSL*. We summarise the results of the previous sections and outline the future work in Section 5.

2 Bilattices

In this section we briefly discuss bilattices and their properties, which will be useful in later sections, where we define bilattice-based language and theory. Notion of a bilattice generalises the Belnap's lattice with four values - *true*, *false*, *none*, *both.* This lattice of Belnap suggested that we can compare facts not only from the point of view of them being true or false, but we can also question how much information about facts is available to us. This gave rise to the notion of a bilattice - a structure with two orderings, usually called *truth* and *knowledge* ordering.
We use the well-known definition of bilattices due to Ginsberg, see [11] or [10].

Definition 1. *A* bilattice $\mathbf{B}$ *is a sextuple* $(\mathbf{B}, \vee, \wedge, \oplus, \otimes, \neg)$ *such that* $(\mathbf{B}, \vee, \wedge)$ *and* $(\mathbf{B}, \oplus, \otimes)$ *are both complete lattices, and* $\neg : \mathbf{B} \rightarrow \mathbf{B}$ *is a mapping satisfying the following three properties:* $\neg^2 = Id_{\mathbf{B}}$*,* $\neg$ *is a dual* *lattice homomorphism from* $(\mathbf{B}, \vee, \wedge)$ *to* $(\mathbf{B}, \wedge, \vee)$*, and* $\neg$ *is the identity mapping on* $(\mathbf{B}, \oplus, \otimes)$*.*

Let $L_1 = (\mathcal{L}_1, \leq_1)$ and $L_2 = (\mathcal{L}_2, \leq_2)$ be two lattices, let x_1, x_2 denote arbitrary elements of the lattice L_1, and let y_1, y_2 denote arbitrary elements of the lattice L_2. Let $\cap_1, \cup_1$ denote the meet and join defined in the lattice L_1, and let $\cap_2, \cup_2$ denote the meet and join defined in the lattice L_2.

Definition 2. *Suppose* $L_1 = (\mathcal{L}_1, \leq_1)$ *and* $L_2 = (\mathcal{L}_2, \leq_2)$ *are complete lattices. Form the set of points* $\mathcal{L}_1 \times \mathcal{L}_2$*, and define the two orderings* $\leq_t$ *and* $\leq_k$ *on* $\mathcal{L}_1 \times \mathcal{L}_2$ *as follows.*
(1) $\langle x_1, y_1\rangle \leq_t \langle x_2, y_2\rangle$ *if and only if* $x_1 \leq_1 x_2$ *and* $y_2 \leq_2 y_1$*.*
(2) $\langle x_1, y_1\rangle \leq_k \langle x_2, y_2\rangle$ *if and only if* $x_1 \leq_1 x_2$ *and* $y_1 \leq_2 y_2$*.*
Therefore, $\langle x_1, y_1\rangle \wedge \langle x_2, y_2\rangle = \langle x_1 \cap_1 x_2, y_1 \cup_2 y_2\rangle$, $\langle x_1, y_1\rangle \vee \langle x_2, y_2\rangle = \langle x_1 \cup_1 x_2, y_1 \cap_2 y_2\rangle$, $\langle x_1, y_1\rangle \otimes \langle x_2, y_2\rangle = \langle x_1 \cap_1 x_2, y_1 \cap_2 y_2\rangle$*, and* $\langle x_1, y_1\rangle \oplus \langle x_2, y_2\rangle = \langle x_1 \cup_1 x_2, y_1 \cup_2 y_2\rangle$*.*

We denote the resulting structure by $L_1 \times L_2 = (\mathcal{L}_1 \times \mathcal{L}_2, \leq_t, \leq_k) = (\mathbf{B}, \leq_t, \leq_k)$, where $\mathbf{B}$ denotes $\mathcal{L}_1 \times \mathcal{L}_2$.
Infinite meet and join with respect to k- and t-orderings will be denoted by $\prod$, $\sum$, $\bigwedge$, $\bigvee$.
From the definition of a lattice, it follows that $\oplus$, $\otimes$, $\vee$ and $\wedge$ are idempotent, commutative, associative, and the absorption laws hold for $\oplus$ and $\otimes$, and $\vee$ and $\wedge$. See, for example, [12] for more details.
The unary operation $\neg$, the complement with respect to the truth ordering, is defined in $L_1 \times L_2$ as follows: $\neg\langle x, y\rangle = \langle y, x\rangle$. This particular definition of negation requires that $L_1 = L_2$, and so we make this assumption throughout. There are some alternative definitions of negation in bilattices, [6, 10], but we do not use them here.

Proposition 1. *[5, 11] Suppose* $\mathbf{B}$ *is a distributive bilattice. Then there are distributive lattices* L_1 *and* L_2 *such that* $\mathbf{B}$ *is isomorphic to* $L_1 \times L_2$*.*

Therefore, we will consider only logic programs over distributive bilattices and regard the underlying bilattice of any program as a product of two lattices. Moreover, we always treat each bilattice we work with as isomorphic to some finite subset of $\mathbf{B} = L_1 \times L_2 = ([0, 1], \leq) \times ([0, 1], \leq)$, where $[0, 1]$ is the unit interval of reals with the linear ordering defined on it. Elements of such a bilattice are pairs: the first element of each pair denotes evidence for a fact, and the second element denotes evidence against it. We assume that each bilattice we work with has four extreme elements, $\langle 0, 0\rangle$ (*none*), $\langle 0, 1\rangle$ (*false*), $\langle 1, 0\rangle$ (*true*) and $\langle 1, 1\rangle$ (*both*).

3 Bilattice-Based Annotated Logic

Now we are ready to introduce Bilattice-Based Annotated Logic. We define its syntax, semantics, and sound and complete sequent calculus for it, see also [19].
We define the bilattice-based annotated language $\mathcal{L}$ to consist of individual variables, individual constants, functions and predicate symbols

together with annotation variables, annotation constants and annotation functions over some fixed bilattice.
Individual variables, constants, and functions are used to build terms, while annotation variables, constants and functions are used to build annotation terms. Typical annotation functions will be $\oplus, \otimes, \wedge, \vee$ taken from a given bilattice.
We allow several connectives to represent operations over a bilattice, and so we allow five connectives and four quantifiers, as follows: $\oplus, \otimes, \wedge, \vee, \neg$ $\Sigma, \Pi, \exists, \forall$.
We use the conventional definition of a first-order term and a first-order formula. An *annotated formula* is defined inductively as follows: if R is an n-ary predicate symbol, $t_1, \ldots, t_n$ are terms, τ is an annotation term, then $R(t_1, \ldots, t_n) : (\tau)$ is an *annotated formula* (called an *annotated atom*). If ψ, ϕ are annotated formulae, then $\psi \odot \phi$ is an annotated formula, where $\odot$ is one of $\{\oplus, \otimes, \wedge, \vee\}$. If F_1 and F_2 are formulae, and τ is an annotation term, then $(F_1 \odot F_2) : (\tau)$ is an annotated formula, where $\odot$ is one of $\{\oplus, \otimes, \wedge, \vee\}$. If ϕ is an annotated formula, then $\neg\phi$, $\Sigma\phi, \Pi\phi, \exists\phi$ and $\forall\phi$ are annotated formulae. In order to reflect the nature of annotations taken from the bilattice, we will alternatively use notation (μ, ν) instead of (τ) when discussing annotation terms.
Note that we use angled brackets $\langle\ \rangle$ to denote elements of $\mathbf{B}$ and round brackets $(\ )$ to denote annotations of the language $\mathcal{L}$.

Example 1. Consider a binary predicate `connected`, which describes the fact of existence of an edge in a probabilistic graph. These graphs can be used to describe the behaviour of internet connections, for example. Then $\texttt{connected}(a, b) : (\frac{1}{3}, \frac{2}{3})$ will describe the fact that the probability of establishing a connection between nodes a and b is equal to $\frac{1}{3}$, while probability of losing this connection is $\frac{2}{3}$. Then, for example, $\texttt{connected}(a, b) : (\frac{1}{3}, \frac{2}{3}) \wedge (\mu, \nu)$ is also a formula which contains a function $\wedge$ and two free variables (μ, ν) in its annotation.

3.1 Interpretations

Let $\mathbf{B}$ denote a bilattice underlying the annotated language $\mathcal{L}$; it will provide the set of truth values for $\mathcal{L}$.
Following the conventional definition of an interpretation (see [3], for example), we fix a domain D, where constants and function symbols receive interpretation. A *variable assignment* V is an assignment, to each variable in $\mathcal{L}$, of an element in the domain D. An interpretation of constants, variables, and function symbols in D will be called a pre-interpretation, we will denote it by $|\ |$. An *interpretation* $\mathcal{I}$ for a (first-order) bilattice-based annotated language $\mathcal{L}$ consists of a pre-interpretation together with a mapping $|R|_{\mathcal{I}} : D^n \longrightarrow \mathbf{B}$ for each n-ary predicate symbol R in $\mathcal{L}$.
One further piece of notation we need is as follows: for each element $\langle\alpha, \beta\rangle$ of $\mathbf{B}$, we denote by $\chi_{\langle\alpha,\beta\rangle} : \mathbf{B} \longrightarrow \mathbf{B}$ the mapping defined by $\chi_{\langle\alpha,\beta\rangle}(\langle\alpha', \beta'\rangle) = \langle 1, 0\rangle$ if $\langle\alpha, \beta\rangle \leq_k \langle\alpha', \beta'\rangle$ and $\chi_{\langle\alpha,\beta\rangle}(\langle\alpha', \beta'\rangle) = \langle 0, 1\rangle$ otherwise.
We will use the two functions $\mathcal{I}$ and χ to define interpretation I for annotated atoms. Given an annotated atom $A : (\alpha', \beta')$ with constant

annotation (α', β'), an interpretation $\mathcal{I}$ for a first-order formula A, and a value $\langle \alpha, \beta \rangle$ from $\mathbf{B}$ assigned to A, we use χ as follows: if the value $\langle \alpha, \beta \rangle \geq_k \langle \alpha', \beta' \rangle$, then $I(A : (\alpha', \beta')) = \langle 1, 0 \rangle$, and $I(A : (\alpha', \beta')) = \langle 0, 1 \rangle$ otherwise. If the annotated term τ attached to an annotated atom $A : \tau$ contains variables $\overline{\mu}, \overline{\nu}$, we use existential quantifier Σ when applying χ as follows: $\chi_\tau(\langle \alpha, \beta \rangle) = \langle 1, 0 \rangle$ if $\Sigma(\overline{\mu}, \overline{\nu})(\tau \geq \langle \alpha, \beta \rangle)$. We will assume this quantification when giving the next definition.

Definition 3. *Let $\mathcal{I}$ be a (first-order) interpretation with domain D for a first order annotated language $\mathcal{L}$ and let V be a variable assignment. Then an annotated formula F in $\mathcal{L}$ can be given an interpretation $I(F)$ (also denoted as $|F|_{I,V}$) in $\mathbf{B}$ as follows.*

- *If F is an annotated atom $R(t_1, \ldots, t_n) : (\mu, \nu)$ with (μ, ν) being an annotation term, then the value of $|F|_{I,V}$ is given by $|F|_{I,V} = \chi_{\langle \mu, \nu \rangle}(|R|_{\mathcal{I}}(|t_1|, \ldots, |t_n|))$.*
- *If F has the form $(\neg A) : (\mu, \nu)$, with A being some first-order atom and (μ, ν) being an annotation term, then $|(\neg A) : (\mu, \nu)|_{I,V} = |A : (\nu, \mu)|_{I,V}$.*
- *If F has the form $\neg(A : (\mu, \nu))$, with A being some first-order formula and (μ, ν) being an annotation term, then $|\neg(A : (\mu, \nu))|_{I,V} = \neg^*(|A : (\mu, \nu)|_{I,V})$, where the operation $\neg^*$ denotes the restriction of the bilattice operation $\neg$ to the set of values $\{\langle 1, 0 \rangle, \langle 0, 1 \rangle\}$.*
- *If F has the form $(F_1 \otimes F_2)$, where F_1 and F_2 are annotated atoms, then $|F_1 \otimes F_2|_{I,V} = |F_1|_{I,V} \otimes |F_2|_{I,V}$.*
  Note that on the left hand side of this equation, the symbol $\otimes$ denotes a connective in $\mathcal{L}$, and on the right hand side it denotes an operation of the bilattice $\mathbf{B}$.
- *If an annotated formula has the form $(A_1 \otimes A_2) : (\mu, \nu)$, where A_1, A_2 are first-order formulae and (μ, ν) being an annotation term, then $|(A_1 \otimes A_2) : (\mu, \nu)|_{I,V} = \chi_{\langle \mu, \nu \rangle}(|A_1|_{\mathcal{I},V} \otimes |A_2|_{\mathcal{I},V})$.*
- *If a formula has the form $\Sigma x R(x) : (\mu, \nu)$, with (μ, ν) being an annotation term, then*

$$|\Sigma x R(x) : (\mu, \nu)|_{I,V} = \chi_{\langle \mu, \nu \rangle} \left(\sum_{d \in D} |R(d)|_{\mathcal{I},V} \right),$$

  where $\sum$ is the infinite join with respect to $\leq_k$, $|R(d)|_{\mathcal{I},V}$ receives interpretation with respect to $\mathcal{I}$ and $V(\frac{x}{d})$, where $V(\frac{x}{d})$ is V except that x is assigned d.

We omit the definitions of interpretations for the remaining connectives and quantifiers, and the reader can easily complete Definition 3. We simply mention here that the connectives $\oplus$, $\otimes$, $\wedge$, $\vee$ and the quantifiers Σ, Π, $\exists$, $\forall$ are interpreted by finite and infinite operations defined on bilattices as in Section 2.
In general, the analogs of the classical truth values true and false are represented by $\langle 1, 0 \rangle$ and $\langle 0, 1 \rangle$ - the greatest and least elements of the bilattice with respect to $\leq_t$. Furthermore, the definitions of satisfiable, unsatisfiable, valid and non-valid formulae and of models are standard if the classical truth values true and false in these definitions are replaced by $\langle 1, 0 \rangle$ and $\langle 0, 1 \rangle$.

The following properties of I are very important for the development of a sequent calculus for BAL.

Proposition 2 (Properties of I). *Let $F, F_1, \dots F_k$ be first-order formulae, and fix a first-order interpretation $\mathcal{I}$ for them. Then any interpretation I built upon $\mathcal{I}$ as in Definition 3 has the following properties:*
1. If $I(F : (\alpha, \beta)) = \langle 1, 0\rangle$, then $I(F : (\alpha', \beta')) = \langle 1, 0\rangle$ for all $\langle \alpha', \beta'\rangle \leq_k \langle \alpha, \beta\rangle$.
2. If $I(F : (\alpha, \beta)) = \langle 0, 1\rangle$, then $I(F : (\alpha', \beta')) = \langle 0, 1\rangle$, for all (α', β') such that $\langle \alpha, \beta\rangle \leq_k \langle \alpha', \beta'\rangle$.
3. $I(F_1 : (\mu_1, \nu_1) \otimes \dots \otimes F_k : (\mu_k, \nu_k)) = \langle 1, 0\rangle \iff I(F_1 : (\mu_1, \nu_1) \oplus \dots \oplus F_k : (\mu_k, \nu_k)) = \langle 1, 0\rangle \iff I(F_1 : (\mu_1, \nu_1) \wedge \dots \wedge F_k : (\mu_k, \nu_k)) = \langle 1, 0\rangle \iff I(F_i : (\mu_i, \nu_i)) = \langle 1, 0\rangle$, for each $i \in \{1, \dots, k\}$.
4. $I(F_1 : (\mu_1, \nu_1) \otimes \dots \otimes F_k : (\mu_k, \nu_k)) = \langle 0, 1\rangle \iff I(F_1 : (\mu_1, \nu_1) \oplus \dots \oplus F_k : (\mu_k, \nu_k)) = \langle 0, 1\rangle \iff I(F_1 : (\mu_1, \nu_1) \vee \dots \vee F_k : (\mu_k, \nu_k)) = \langle 0, 1\rangle \iff I(F_i : (\mu_i, \nu_i)) = \langle 0, 1\rangle$, for each $i \in \{1, \dots, k\}$.
5. If $I(F_1 : (\mu_1, \nu_1) \odot \dots \odot F_k : (\mu_k, \nu_k)) = \langle 1, 0\rangle$, then $I((F_1 \odot \dots \odot F_k) : ((\mu_1, \nu_1) \odot \dots \odot (\mu_k, \nu_k))) = \langle 1, 0\rangle$, where $\odot$ is any one of the connectives $\otimes, \oplus, \wedge$.
6. If $I(F_1 : (\mu_1, \nu_1) \odot \dots \odot F_k : (\mu_k, \nu_k)) = \langle 0, 1\rangle$, then $I((F_1 \odot \dots \odot F_k) : ((\mu_1, \nu_1) \odot \dots \odot (\mu_k, \nu_k))) = \langle 0, 1\rangle$, where $\odot$ is any one of the connectives $\otimes, \oplus, \vee$.
7. If $I(F_1 : (\mu, \nu)) = \langle 1, 0\rangle$, then $I((F_1 \oplus F_2) : (\mu, \nu)) = \langle 1, 0\rangle$, for any formula F_2.
8. If $I(F_1 : (\mu, \nu)) = \langle 0, 1\rangle$, then $I((F_1 \otimes F_2) : (\mu, \nu)) = \langle 0, 1\rangle$, for any formula F_2.
9. $I(F : (\mu, \nu)) = \langle 1, 0\rangle \iff I(\neg F : (\nu, \mu)) = \langle 1, 0\rangle$.
10. $I(F : (\mu, \nu)) = \langle 0, 1\rangle \iff I(\neg F : (\nu, \mu)) = \langle 0, 1\rangle$.
11. For every formula F, $I(F : (0, 0)) = \langle 1, 0\rangle$.

The proof of this proposition uses the definitions of I and bilattice operations. Not all the statements in the above proposition are immediate. For example, items $3 - 6$ exhibit some non-trivial properties of the bilattice interpretation. The proposition will be useful in the next subsection, where we introduce a sound and complete sequent calculus for BAL.

3.2 Sequent Calculus for Annotated Logics.

In this section we will use properties of bilattices and interpretation I to define a sequent calculus for Bilattice-Based Annotated Logic (BAL).
We denote annotated formulae by φ, ψ and ϕ, and conventional first-order formulae by F_1, F_2. The symbols Ω and Γ will denote arbitrary finite sequences of annotated formulae.
We allow the following axioms $\varphi \mapsto \varphi$; $\mapsto F : (0, 0)$ and rules:
1. Introducing $\neg$:

$$\frac{\Omega \mapsto \Gamma, \varphi}{\neg\varphi, \Omega \mapsto \Gamma,}\ \neg\text{L}\ , \qquad \frac{\varphi, \Omega \mapsto \Gamma}{\Omega \mapsto \Gamma, \neg\varphi}\ \neg\text{R}.$$

2. Introducing $\vee$:

$$\frac{\psi, \Omega \mapsto \Gamma;\ \phi, \Omega \mapsto \Gamma}{\phi \vee \psi, \Omega \mapsto \Gamma}\ \vee\text{-L}, \qquad \frac{\Omega \mapsto \Gamma, \varphi}{\Omega \mapsto \Gamma, \varphi \vee \psi}\ \text{or}\ \frac{\Omega \mapsto \Gamma, \psi}{\Omega \mapsto \Gamma, \varphi \vee \psi}\ \vee\text{-R}.$$

3. Introducing $\wedge$:

$$\frac{\psi, \Omega \mapsto \Gamma}{\psi \wedge \phi, \Omega \mapsto \Gamma} \text{ or } \frac{\phi, \Omega \mapsto \Gamma}{\psi \wedge \phi, \Omega \mapsto \Gamma} \wedge\text{-L}, \qquad \frac{\Omega \mapsto \Gamma, \varphi \ ; \ \Omega \mapsto \Gamma, \psi}{\Omega \mapsto \Gamma, \varphi \wedge \psi} \wedge\text{-R}.$$

4. Introducing $\oplus$:

$$\frac{\psi, \Omega \mapsto \Gamma; \ \phi, \Omega \mapsto \Gamma}{\psi \oplus \phi, \Omega \mapsto \Gamma} \oplus\text{-L}, \qquad \frac{\Omega \mapsto \Gamma, \varphi \ ; \ \Omega \mapsto \Gamma, \psi}{\Omega \mapsto \Gamma, \varphi \oplus \psi} \oplus\text{-R}.$$

The rule $\oplus$-L is identical to $\vee$-L; but the rule $\oplus$-R is identical to $\wedge$-R. Cf. Definition 2 and Proposition 2.(3-4).
5. Introducing $\otimes$:

$$\frac{\psi, \Omega \mapsto \Gamma; \ \varphi, \Omega \mapsto \Gamma}{\psi \otimes \varphi, \Omega \mapsto \Gamma} \otimes\text{-L}, \qquad \frac{\Omega \mapsto \Gamma, \varphi \ ; \ \Omega \mapsto \Gamma, \psi}{\Omega \mapsto \Gamma, \varphi \otimes \psi} \otimes\text{-R}.$$

The rule $\otimes$-R is identical to $\wedge$-R; but $\otimes$-L is identical to $\vee$-L. Cf. Definition 2 and Proposition 2.(3-4).
Similarly, because quantifiers Π, Σ are modelled by infinite analogues of $\otimes$ and $\oplus$, rules for introducing Π are identical to the rules for Σ:
6. Introducing $\bigcirc = \Sigma, \Pi$ in the antecedent (Σ-L/ Π-L):

$$\frac{S^{y_i}_{x_i}\varphi, \Omega, \mapsto \Gamma}{\bigcirc x_i \varphi, \Omega \mapsto \Gamma}\ , y_i \notin FREE(\Omega \cup \{\Sigma x_i \varphi, \Gamma\}).$$

7. Introducing $\bigcirc = \Sigma, \Pi$ in the consequent (Σ-R/ Π-R):

$$\frac{\Omega \mapsto \Gamma, S^{y_i}_{x_i}\varphi}{\Omega \mapsto \Gamma, \bigcirc x_i \varphi}\ , y_i \notin FREE(\Omega \cup \{\Sigma x_i \varphi, \Gamma\}).$$

10. Introducing $\exists$ in the antecedent ($\exists$-L):

$$\frac{S^{y_i}_{x_i}\varphi, \Omega \mapsto \Gamma}{\exists x_i \varphi, \Omega \mapsto \Gamma}\ , y_i \notin FREE(\Omega \cup \{\exists x_i \varphi, \Gamma\}).$$

11. Introducing $\exists$ in the consequent ($\exists$-R):

$$\frac{\Omega \mapsto \Gamma, S^{t}_{x_i}\varphi}{\Omega \mapsto \Gamma, \exists x_i \varphi}\ .$$

The rules for quantifier $\forall$ are given dually to those for $\exists$, as in conventional classical sequent calculi.
The next block of rules will give an account of annotations:
12. Increasing values in annotations (IA), (*cf. Proposition 2.(1)*):

$$\frac{\Omega \mapsto \Gamma, F : (\mu, \nu)}{\Omega \mapsto \Gamma, F : (\mu', \nu')}\ , \text{ for all } (\mu', \nu') \leq_k (\mu, \nu).$$

13. Descending values in annotations (DA) (*cf. Proposition 2.(2)*):

$$\frac{F : (\mu, \nu), \Omega \mapsto \Gamma}{F : (\mu', \nu'), \Omega \mapsto \Gamma}\ , \text{ for all } (\mu, \nu) \leq_k (\mu', \nu').$$

14. Introducing $\oplus$ inside annotated formulae in the antecedent (I$\oplus$I -L) (*cf. Proposition 2.(7)*):

$$\frac{F_1 : (\mu, \nu), \Omega \mapsto \Gamma; \ F_2 : (\mu, \nu), \Omega \mapsto \Gamma}{(F_1 \oplus F_2) : (\mu, \nu), \Omega \mapsto \Gamma}\ .$$

15. Introducing $\oplus$ inside annotated formulae in the consequent ($\oplus$I-R) (*cf. Proposition 2.(7)*):

$$\frac{\Omega \mapsto \Gamma, F_1 : (\mu, \nu)}{\Omega \mapsto \Gamma, (F_1 \oplus F_2) : (\mu, \nu)}, \qquad \frac{\Omega \mapsto \Gamma, F_2 : (\mu, \nu)}{\Omega \mapsto \Gamma, (F_1 \oplus F_2) : (\mu, \nu)} .$$

16. Introducing $\otimes$ inside annotated formulae in the antecedent ($\otimes$I-L) (*cf. Proposition 2.(8)*):

$$\frac{F_1 : (\mu, \nu), \Omega \mapsto \Gamma}{(F_1 \otimes F_2) : (\mu, \nu), \Omega \mapsto \Gamma}, \qquad \frac{F_2 : (\mu, \nu), \Omega \mapsto \Gamma}{(F_1 \otimes F_2) : (\mu, \nu), \Omega \mapsto \Gamma} .$$

17. Introducing $\otimes$ inside annotated formulae in the consequent (I $\otimes$ I-R) (*cf. Proposition 2.(8)*):

$$\frac{\Omega \mapsto \Gamma, F_1 : (\mu, \nu); \;\; \Omega \mapsto \Gamma, F_2 : (\mu, \nu)}{\Omega \mapsto \Gamma, (F_1 \otimes F_2) : (\mu, \nu)} .$$

18. Combining annotations in the antecedent (CAL) (*cf. Proposition 2.(6)*):

$$\frac{F_1 : (\mu_1, \nu_1) \odot F_2 : (\mu_2, \nu_2), \Omega \mapsto \Gamma}{(F_1 \odot F_2) : ((\mu_1, \nu_1) \odot (\mu_2, \nu_2)), \Omega \mapsto \Gamma}, \text{ where } \odot \text{ is either } \oplus, \otimes \text{ or } \vee.$$

19. Combining annotations in the consequent (CAR) (*cf. Proposition 2.(5)*, see also Example 2):

$$\frac{\Omega \mapsto \Gamma, F_1 : (\mu_1, \nu_1) \odot F_2 : (\mu_2, \nu_2)}{\Omega \mapsto \Gamma, (F_1 \odot F_2) : ((\mu_1, \nu_1) \odot (\mu_2, \nu_2))}, \text{ where } \odot \text{ is either } \oplus, \otimes \text{ or } \wedge.$$

20. Introducing $\neg$ inside annotations (*cf. Proposition 2.(11)*):

$$\frac{F_1 : (\nu, \mu), \Omega \mapsto \Gamma}{(\neg F_1) : (\mu, \nu), \Omega \mapsto \Gamma}\neg\text{I-L}, \qquad \frac{\Omega, \mapsto \Gamma, F_1 : (\nu, \mu)}{\Omega \mapsto \Gamma, (\neg F_1) : (\mu, \nu)}\neg\text{I-R} .$$

We allow the structural rules interchange, contraction and weakening. These structural rules can be defined to be either primitive or admissible, in style of **G3**. The latter option seems to be more appropriate for automated reasoning, but we shall not discuss this issue here.
All the annotation functions $\otimes$, $\oplus$, $\wedge$ and $\vee$ are defined in **B**, and one is allowed to operate with them accordingly. That is, for example, one can think of $F : ((\frac{1}{3}, \frac{2}{3}) \otimes (\frac{2}{3}, \frac{1}{3}))$ as of $F : (\frac{1}{3}, \frac{1}{3})$. Also, because in both lattices constituting a bilattice, operations $\oplus$, $\otimes$, $\wedge$ and $\vee$ are idempotent, commutative, associative, and distributive, one can treat equally $F : (\mu, \nu)$ and $F : ((\mu, \nu) \odot (\mu, \nu))$ ($\odot = \oplus, \otimes, \wedge, \vee$), for example, or write $F : ((\mu_1, \nu_1) \vee ((\mu_2, \nu_2) \wedge (\mu_3, \nu_3)))$ instead of $F : (((\mu_1, \nu_1) \vee (\mu_2, \nu_2)) \wedge ((\mu_1, \nu_1) \vee (\mu_3, \nu_3)))$, and so on. All these properties can be stated directly as sequent rules, or, as we do here, just assumed throughout. In fact, the latter way seems to be more natural: $\otimes, \oplus, \wedge$ and $\vee$ appearing in annotations are not connectives, but they are annotation functions. And, in the same way as we ignore properties of functions appearing in individual terms when defining conventional sequent rules, we may wish to ignore properties of annotation functions when defining sequent calculus for BAL.
In general, all the classical tautologies reformulated with annotated formulae are provable in this calculus. But additionally, we can prove theorems concerning properties of annotations.

Example 2. The formula $(F_1 \wedge F_2) : (0,1)$ is a logical consequence of $F_1 : (1,0) \wedge F_2 : (0,1)$. One can use the rule CAR and the fact that annotation terms $(1,0) \wedge (0,1)$ and $(0,1)$ are equal to prove this sequentially.

Cut elimination theorem can be proven for BAL, if cut rule is defined. But we will not discuss this issue here.

Theorem 1 (Soundness). *For any annotated formula φ, if $\psi \vdash \varphi$ then $\psi \models \varphi$.*

Proof. Proof follows along the lines of conventional proof of soundness for classical first-order sequent calculus. We additionally make use of Proposition 2.

The calculus for BAL is also complete. But we avoid to state Completeness until the last section, when we obtain it as a corollary of the completeness theorem for many-sorted logic.

4 Many-Sorted Representation of BAL

The sequent calculus for BAL introduced in the previous section reflects rigorously the semantic properties of the logic. But it may be criticised for having a difficult rule representation and allowing semantics to interfere with its syntax. In the second part of this paper we will show how this calculus can equivalently be transformed into the elegant conventional sequent calculus for many-sorted logic, introduced in [4]. The latter is proven to be sound and complete, see, for example [4, 23].
In this section we draw inspiration from the work of Manzano [23] who showed how second-order and dynamic logic can be translated into fragments of many-sorted logic MSL. We define the syntax and semantics of its fragment MSL* and use it for translation of BAL.
We follow the notation of Manzano when working with many-sorted semantics. We hope that the uniformity of our notation with that of [23] will make it easier to consider our results in one context with the similar results of Manzano concerning second-order and dynamic logic.

4.1 Many-Sorted Language MSL* of Signature Σ^*

We start by defining the many-sorted language MSL* of signature Σ^*.

Definition 4. *We define a* signature $\Sigma^* = \langle \mathrm{SORT}, \mathrm{RANK} \rangle$*, where*

$\mathrm{SORT}(\Sigma^*) = \mathrm{SORT} = \{0, 1, 2, < 0, \overbrace{1, \ldots, 1}^{n-1}, 2 >\}$ *(representing boolean, individual, bilattice universes and a universe of n-ary relations on individuals and bilattice elements); and* RANK *is described in Df. 5.*

We will continue defining RANK in the next Definition, and will pause for a while to explain the significance of the particular choice of SORT for MSL*. The sorts 2 and $< 0, \overbrace{1, \ldots, 1}^{n-1}, 2 >$ are specific for MSL*, if

we compare them with the general definition of a many-sorted signature in [23]. The sort 2 represents universe of bilattice elements. The sort $< 0, \overbrace{1, \ldots, 1}^{n-1}, 2 >$ will be used when we give a formal account of the interpretation function I for BAL, which will be formalised by relations I^k of different arities in MSL*. As in second-order logic, the relations interpreted in this universe will be quantified. Moreover, they will have both individual and annotation terms as arguments. As any other relations, they can be evaluated as true, if some elements in the underlying structure satisfy this relation, or false otherwise.
We define S_ω(SORT) to be the set of all finite sequences of elements of SORT.
RANK is a function whose values are in S_ω(SORT).
We denote Dom(RANK) as OPER.SYM(Σ^*) = OPER.SYM and call its elements operation symbols. We allow the following operation symbols in the language:

- $\neg, \vee, \wedge$ - the classical connectives;
- $f_1^n, f_2^n, \ldots$, for any $n \in \mathbf{N}$, - function symbols of different arities over individual terms;
- $\vartheta_1^n, \vartheta_2^n, \ldots$, for any $n \in \mathbf{N}$, - bilattice function symbols of different arities;
- $R_1^n, R_2^n, \ldots$, for any $n \in \mathbf{N}$, - predicates over individual terms;
- $\leq_k$ - bilattice binary relation;
- $I_1^n, I_2^n, \ldots,$, for any $n \in \mathbf{N}$, - relation symbols of different arities;
- ε - membership relation.

Then RANK is defined for each of them as follows.

Definition 5 (Df. 4 continued).
For $\neg, \vee, \wedge \in$ OPER.SYM, RANK($\vee$) = RANK($\wedge$) = $\langle 0, 0, 0 \rangle$, RANK($\neg$) = $\langle 0, 0 \rangle$.

We define RANK$(f^n(x_1, \ldots, x_n)) = \langle 1, \overbrace{1, \ldots, 1}^{n} \rangle$, *where f^n is n-ary function over terms of sort* 1*; and*

RANK$(\vartheta^n((\mu_1, \nu_1), \ldots, (\mu_n, \nu_n))) = \langle 2, \overbrace{2, \ldots, 2}^{n} \rangle$, *where ϑ^n is n-ary function over terms of sort* 2*;*

RANK$(R^n(x_1, \ldots, x_n)) = \langle 0, \overbrace{1, \ldots, 1}^{n} \rangle$, *where R^n is n-ary relation over individual terms of sort* 1*;*
For the binary bilattice relation $\leq_k$, RANK($\leq_k$) = $\langle 0, 2, 2 \rangle$.

RANK$(I^n((x_1, \ldots, x_n), (\mu, \nu))) = \langle 0, \overbrace{1, \ldots, 1}^{n}, 2 \rangle$, *where I^n is an n-ary relation over terms of sorts* 1 *and* 2.
And, finally, for membership relations ε_n,

$$\mathrm{RANK}(\varepsilon_n) = \langle 0, \overbrace{1, \ldots, 1}^{n}, 2 < 0, \overbrace{1, \ldots, 1}^{n}, 2 > \rangle.$$

Note that each annotation term of the form (μ_i, ν_i) is interpreted by one element of a bilattice, and thus each annotation term has a single sort 2, and not $(2, 2)$ as one might expect.

Definition 6. *We define a many-sorted* structure

$$S = \langle A_1, A_2, A_3^n, f^{A_1}, f^{A_2}\rangle, (\textit{for each } n \in \mathbb{N}),$$

where A_1, A_2 and A_3^n are universes for variables of sorts $< 1 >$, $< 2 >$, $< 0, \overbrace{1, \ldots, 1}^{n-1}, 2 >$; $f^{A_1} \subseteq A_1^k$ and $f^{A_2} \subseteq A_2^k$.

A many-sorted language $\mathcal{L}$ consists of symbols from OPER.SYM, quantifier $\exists$ and the set of variables $\mathcal{V} = V^i : i \in \text{SORT} - \{0\}$. That is, the superscript of a variable denotes its sort.

Definition 7. *Expressions of the language are defined inductively as follows:*

1. *Each variable of a sort i is an expression of the same type.*
2. *If $f^m \in$ OPER.SYM and $\epsilon_1, \ldots, \epsilon_m$ are expressions of the single type 1, then $f^m(\epsilon_1, \ldots, \epsilon_m)$ is an expression of type 1.*
   If $\vartheta^m \in$ OPER.SYM and $\epsilon_1, \ldots, \epsilon_m$ are expressions of type 2, then $\vartheta(\epsilon_1, \ldots, \epsilon_m)$ is an expression of type 2.
   If $R^m \in$ OPER.SYM , then for all the expressions ϵ of the single type 1, the string $R(\epsilon_1, \ldots, \epsilon_m)$ is an expression of type 0.
   If $\leq_k \in$ OPER.SYM, then for all expressions ϵ_1, ϵ_2 of the single type 2, the string $\epsilon_1 \leq_k \epsilon_2$ is an expression of type 0.
   If $I^{m+1} \in$ OPER.SYM, $\epsilon_1, \ldots, \epsilon_m$ are expressions of the single type 1 and ϵ is an expression of the single type 2, then $I^{m+1}((\epsilon_1, \ldots, \epsilon_m), \epsilon)$ is an expression of type 0.
3. *If ϵ is an expression of type 0 and x^i is a variable of sort i, then $\exists x^i \epsilon$ is an expression of type 0 as well.*

No other string is an expression.

Terms are expressions of single non-zero type $i = 1, 2$. Formulae are expressions of type 0.
Note that in our setting, the relations I of different arities can be viewed as variables and can be quantified. This is why, we handle these relations uniformly with the way how [23] treated second order relations and formulae within many-sorted language. We require $\varepsilon t_1, \ldots, t_n, \mu, \nu I^n$, with ε being the membership relation, to replace $I^n((t_1, \ldots, t_n), (\mu, \nu))$; and we define expressions of the former type to be formulae, but expressions of the latter type - not, and amend Definition 7 accordingly.
We define $\supset$ and $\forall$ using $\neg$, $\vee$ and $\exists$ in the usual way.

Interpretation We define the interpretation function **I** for the many-sorted language, following closely [23], but making a substantial adaptation to the particular language MSL* we have defined.
We define Assignment

$$M : \bigcup \underset{i \in \text{SORT}-\{0\}}{\mathcal{V}_i} \rightarrow \underset{i \in \text{SORT}-\{0\}}{A_i},$$

in such a way that $M[\mathcal{V}_i] \subseteq A_i$.

Definition 8. *An interpretation **I** over a structure S is a pair $\langle S, M\rangle$, where M is an assignment on S. In particular, for $i \in$ SORT,*

1. $\mathbf{I}(x^i) = M(x^i)$,
2. $\mathbf{I}(f(\epsilon_1, \ldots, \epsilon_n)) = f^S(\mathbf{I}(\epsilon_1), \ldots, \mathbf{I}(\epsilon_n))$.
As particular cases of item 2, we have:
$\mathbf{I}(a^i) = (a^i)^S$;
3. $\mathbf{I}(f(t_1, \ldots, t_n)) = (f)^S(\mathbf{I}(t_1), \ldots, \mathbf{I}(t_n))$;
4. $\mathbf{I}(\vartheta(\tau_1, \ldots, \tau_n)) = (\vartheta)^S(\mathbf{I}(\tau_1), \ldots, \mathbf{I}(\tau_n))$;
$\mathbf{I}(R(t_1, \ldots, t_n)) = (R)^S(\mathbf{I}(t_1), \ldots, \mathbf{I}(t_n))$;
$\mathbf{I}((t_1, \ldots, t_n), (\tau)\varepsilon I) = \varepsilon^S\langle(\mathbf{I}(t_1), \ldots, \mathbf{I}(t_n), \mathbf{I}(\tau)), ((I)^S)\rangle$;
$\mathbf{I}(\neg\psi) = \neg^S(\mathbf{I}(\psi))$;
$\mathbf{I}(\psi \vee \phi) = (\mathbf{I}(\psi)) \vee^S (\mathbf{I}(\phi))$; $\mathbf{I}(\psi \wedge \phi) = (\mathbf{I}(\psi)) \wedge^S (\mathbf{I}(\phi))$.
5. $\mathbf{I}(\exists x^i \phi) =$ *True if and only if* $\{x^i \in A_i | \mathbf{I}^{x^j}_{x^i}(\phi) = \mathit{True}\} \neq \emptyset$ *(where* $\mathbf{I}^{x^j}_{x^i} = \langle S, M^{x^j}_{x^i}\rangle$ *and* $M^{x^j}_{x^i} = (M - \{< x^i, M(x^i) >\}) \cup \{< x^i, x^j >\}$*).*

Note that the interpretation of MSL* is two-valued: atomic formulae are interpreted as true if the relations they formalise are satisfied in the structure S, and they are interpreted false otherwise. Complex formulae are interpreted respective to this interpretation of atomic formulae.
Now, when the many-sorted language MSL*, its underlying structure, and the interpretation function $\mathbf{I}$ are defined, we will show that BAL can be translated into MSL*, and so we give syntactical and semantical translation for BAL.

4.2 Translation of BAL into MSL*

In the subsequent sections we lighten the notation, and instead of using upper indices to denote sorts of variables, will use symbols x and y with lower indices to denote variables of sort 1, and μ, ν with lower indices to denote variables of sort 2.

Syntactical Translation The syntactical translation from BAL to MSL* leaves all individual terms and atomic non-annotated formulae as they are, and gives the following translation to the atomic annotated formulae: $\mathrm{TRANSL}_{BAL\mapsto \mathrm{MSL}^*}(R(\overline{x}) : (\mu, \nu)) =$
$\forall I \exists(\mu', \nu')((R(\overline{x}) \wedge (\overline{x}, (\mu', \nu'))\varepsilon I) \wedge ((\mu, \nu) \leq_k (\mu', \nu')))$.
We will abbreviate $\mathrm{TRANSL}_{BAL\mapsto \mathrm{MSL}^*}$ as TRANSL*.

$$\mathrm{TRANSL}^*(\neg(R(\overline{x}) : (\mu, \nu))) = \mathrm{TRANSL}^*(R(\overline{x}) : (\nu, \mu)).$$

Let ψ_1 and ψ_2 be annotated atomic formulae. Then

$$\mathrm{TRANSL}^*(\neg\psi_1) = \neg\mathrm{TRANSL}^*(\psi_1);$$
$$\mathrm{TRANSL}^*(\psi_1 \wedge \psi_2) = \mathrm{TRANSL}^*(\psi_1) \wedge \mathrm{TRANSL}^*(\psi_2);$$
$$\mathrm{TRANSL}^*(\psi_1 \vee \psi_2) = \mathrm{TRANSL}^*(\psi_1) \vee \mathrm{TRANSL}^*(\psi_2);$$

Because our final goal is to translate sequent calculus for BAL into the conventional sequent calculus for many-sorted logics, the translation function TRANSL* is sensitive to the position of a translated formula in a sequent:

$$\mathrm{TRANSL}^*(\psi_1 \otimes \psi_2) = \mathrm{TRANSL}^*(\psi_1) \wedge \mathrm{TRANSL}^*(\psi_2),$$

if $\psi_1 \otimes \psi_2$ appears in the consequent of a sequent; and

$$\text{TRANSL}^*(\psi_1 \otimes \psi_2) = \text{TRANSL}^*(\psi_1) \vee \text{TRANSL}^*(\psi_2),$$

if $\psi_1 \otimes \psi_2$ appears in the consequent of a sequent.

$$\text{TRANSL}^*(\psi_1 \oplus \psi_2) = \text{TRANSL}^*(\psi_1) \wedge \text{TRANSL}^*(\psi_2),$$

if $\psi_1 \otimes \psi_2$ appears in the antecedent of a sequent; and

$$\text{TRANSL}^*(\psi_1 \oplus \psi_2) = \text{TRANSL}^*(\psi_1) \vee \text{TRANSL}^*(\psi_2),$$

if $\psi_1 \otimes \psi_2$ appears in the antecedent of a sequent.
For complex formulae under a single annotation we introduce the following translation:
$\text{TRANSL}^*(R_1(\overline{x}) \odot R_2(\overline{y}) : (\alpha, \beta)) = \forall I[\exists(\mu_1, \nu_1), (\mu_2, \nu_2)(R_1(\overline{x}) \wedge R_2(\overline{y})$ $\wedge(\overline{x}, (\mu_1, \nu_1)\varepsilon I) \wedge (\overline{y}, (\mu_2, \nu_2)\varepsilon I) \wedge ((\alpha, \beta) \leq_k ((\mu_1, \nu_1) \odot (\mu_2, \nu_2))))]$, for $\odot = \oplus, \otimes, \vee, \wedge$.
Finally, we give a translation for the existential quantifiers:

$$\text{TRANSL}^*(\exists x \psi) = \exists x \text{TRANSL}^*(\psi).$$

$\text{TRANSL}^*(\Sigma x \psi) = \exists x \text{TRANSL}^*(\psi)$, if $\Sigma x \psi$ is in the antecedent of a sequent; and $\text{TRANSL}^*(\Sigma x \psi) = \forall x \text{TRANSL}^*(\psi)$, if $\Sigma x \psi$ is in the consequent of a sequent.

Example 3. The ground formula $(F_1 \wedge F_2) : (0, 1)$ from Example 2 can be translated into $\forall I[\exists(\mu_1, \nu_1), (\mu_2, \nu_2)(F_1 \wedge F_2 \wedge ((\mu_1, \nu_1)\varepsilon I) \wedge ((\mu_2, \nu_2)\varepsilon I) \wedge ((0, 1) \leq_k ((\mu_1, \nu_1) \wedge (\mu_2, \nu_2))))]$.

Semantical translation. We are ready now to compare model properties of BAL and MSL*.

Lemma 1. *Let F be an annotated formula of BAL. Let Σ^* and S be a signature respectively a structure of* MSL**, with* **I** *being a many-sorted interpretation in S. And let $|\;|_I$ be an interpretation for BAL as defined in Section 3. Then the following holds:*

$$|F|_I = \langle 1, 0 \rangle \text{ in BAL} \iff \mathbf{I}(\text{TRANSL}^*(F)) = \texttt{True} \text{ in MSL}^* .$$

Proof. The proof proceeds by two routine induction arguments on complexity of formulae in BAL and MSL*; we use definitions of I in BAL and interpretation **I** over the many-sorted structure S.

Corollary 1. $\models F$ *in* BAL *if and only if* $\models \text{TRANSL}^*(F)$ *in* MSL*.

We have given the syntactical and semantical translation of BAL into MSL*. It remains to show their deductive equivalence.

4.3 Sequent Calculus for MSL*

We define a theory which will be proven to be deductively equivalent to BAL. In fact, sequent calculus for MSL* is just a conventional many-sorted sequent calculus MSL of [4, 23], but with several simple rules added in order to reflect particular properties of bilattice structures[1]:
We add $\mapsto (\mu, \nu) \geq_k (0, 0)$ to the set of axioms.
The rules we add to MSL are:

1. Transitivity of $\leq_k$ in the consequent:
$$\frac{\Omega \mapsto \Gamma, (\mu_1, \nu_1) \leq_k (\mu_2, \nu_2), (\mu_2, \nu_2) \leq_k (\mu_3, \nu_3)}{\Omega \mapsto \Gamma, (\mu_1, \nu_1) \leq_k (\mu_3, \nu_3)} \text{ Tr-R.}$$
2. Transitivity of $\leq_k$ in the antecedent:
$$\frac{(\mu_2, \nu_2) \leq_k (\mu_1, \nu_1), (\mu_3, \nu_3) \leq_k (\mu_2, \nu_2), \Omega \mapsto \Gamma}{(\mu_3, \nu_3) \leq_k (\mu_1, \nu_1), \Omega \mapsto \Gamma} \text{ TR-L.}$$
3. Introduction of $\oplus$ in the antecedent:
$$\frac{(\mu, \nu) \leq_k (\mu_1, \nu_1), \Omega \mapsto \Gamma; \;\; (\mu, \nu) \leq_k (\mu_2, \nu_2), \Omega \mapsto \Gamma}{(\mu, \nu) \leq ((\mu_1, \nu_1) \oplus (\mu_2, \nu_2)), \Omega \mapsto \Gamma} \oplus\text{-L.}$$
4. Introduction of $\oplus$ in the consequent:
$$\frac{\Omega \mapsto \Gamma, (\mu, \nu) \leq_k (\mu_1, \nu_1)}{\Omega \mapsto \Gamma, (\mu, \nu) \leq_k ((\mu_1, \nu_1) \oplus (\mu_2, \nu_2))} \oplus\text{-R.}$$
5. Introduction of $\otimes$ in the antecedent:
$$\frac{(\mu_1, \nu_1) \leq_k (\mu, \nu), \Omega \mapsto \Gamma}{((\mu_1, \nu_1) \otimes (\mu_2, \nu_2)) \leq_k (\mu, \nu), \Omega \mapsto \Gamma} \otimes\text{-L.}$$
6. Introduction of $\oplus$ in the consequent:
$$\frac{\Omega \mapsto \Gamma, (\mu, \nu) \leq_k (\mu_1, \nu_1); \;\; \Omega \mapsto \Gamma, (\mu, \nu) \leq_k (\mu_2, \nu_2)}{\Omega \mapsto \Gamma, (\mu, \nu) \leq_k ((\mu_1, \nu_1) \otimes (\mu_2, \nu_2))}, \; \otimes\text{-R.}$$
7. Introduction of $\otimes$, $\oplus$, $\vee$ in the antecedent:
$$\frac{(\mu_1, \nu_1) \leq_k (\mu', \nu'), \;\; (\mu_2, \nu_2) \leq_k (\mu'', \nu''), \Omega \mapsto \Gamma}{((\mu_1, \mu_2) \odot (\mu_2, \nu_2)) \leq_k ((\mu', \nu') \odot (\mu'', \nu'')), \Omega \mapsto \Gamma} \odot\text{-L,}$$
where $\odot$ is one of $\otimes$, $\oplus$, $\vee$.
8. Introduction of $\otimes$, $\oplus$, $\wedge$ in the consequent:
$$\frac{\Omega \mapsto \Gamma, (\mu_1, \nu_1) \leq_k (\mu', \nu'), (\mu_2, \nu_2) \leq_k (\mu'', \nu'')}{\Omega \mapsto \Gamma, ((\mu_1, \mu_2) \odot (\mu_2, \nu_2)) \leq_k ((\mu', \nu') \odot (\mu'', \nu''))} \odot\text{-R,}$$
where $\odot$ is one of $\otimes$, $\oplus$, $\wedge$.

All the properties of bilattice operations we assumed when working with sequent calculus for BAL are assumed here too. For example, $(\frac{1}{3}, \frac{2}{3}) \otimes (\frac{2}{3}, \frac{1}{3})$ can be substituted by $(\frac{1}{3}, \frac{1}{3})$ throughout the proof. This includes all the lattice axioms and distributivity.

[1] We use a multi succedent reformulation of a single succedent calculus of [4, 23].

Example 4. A many-sorted version of Example 2 can be proven using the sequent rules of MSL and the rule 8 above. That is, we can prove that $\forall I[\exists(\mu_1,\nu_1),(\mu_2,\nu_2)(F_1 \wedge F_2 \wedge ((\mu_1,\nu_1)\varepsilon I) \wedge ((\mu_2,\nu_2)\varepsilon I) \wedge ((0,1) \leq_k ((\mu_1,\nu_1) \wedge (\mu_2,\nu_2))))]$ follows from $[\forall I \exists(\mu_1,\nu_1)(F_1 \wedge ((\mu_1,\nu_1)\varepsilon I) \wedge ((1,0) \leq_k (\mu_1,\nu_1))] \wedge [\forall I \exists(\mu_2,\nu_2)(F_2 \wedge ((\mu_2,\nu_2)\varepsilon I) \wedge ((0,1) \leq_k (\mu_2,\nu_2))]$.

Now we can prove that BAL and MSL* are deductively equivalent.

Theorem 2. *For any annotated formula φ the following holds:*

$$\vdash_{\mathrm{BAL}} \varphi \textit{ iff } \vdash_{\mathrm{MSL}^*} \mathrm{TRANSL}^*(\varphi).$$

Proof. The proof that $\vdash_{\mathrm{MSL}^*} \mathrm{TRANSL}^*(\varphi)$ implies $\vdash_{\mathrm{BAL}} \varphi$ is trivial. The "if" part of the proof proceeds by considering translations of axioms of BAL into proofs in MSL*, and rules for BAL into proofs in MSL*. Namely, for each rule in BAL, we translate the lower sequent of this rule into MSL*, and then show how the translation of the upper sequent(s) of the BAL rule can be derived in MSL*. For example, we take the rule DA from the sequent calculus for BAL. We fix F from the rule to be $R(\overline{x})$. The lower sequent of this rule can be translated into $\forall I \exists(\mu'',\nu'')(R(\overline{x}) \wedge I(\overline{x},(\mu'',\nu'')) \wedge (\mu',\nu') \leq_k (\mu'',\nu''), \Omega \mapsto \varphi$. We also assume that $((\mu,\nu) \leq_k (\mu',\nu'))$ is added to the right hand side of each sequent: this is needed because the condition $((\mu,\nu) \leq_k (\mu',\nu'))$ is attached to the rule DA in BAL. Being put into antecedent of a lower sequent, this translated formula will receive a derivation from the three upper sequents: $\Omega, R(\overline{x}) \mapsto \varphi$; $(\overline{x},(\mu'',\nu'')\varepsilon I), \Omega \mapsto \varphi$; and $(\mu,\nu) \leq_k (\mu'',\nu''), \Omega \mapsto \varphi$. To obtain this, one would need to apply, one after another, MSL* rules $\forall$-L, $\exists$-L, $\wedge$-L, and Tr-L. The three upper sequents will also give a proof for $\forall I \exists(\mu'',\nu'')((R(\overline{x}) \wedge I(\overline{x},(\mu'',\nu'')) \wedge (\mu,\nu) \leq_k (\mu'',\nu''), \Omega \mapsto \varphi$, that is, the translation of the upper sequent of the DA rule $R(\overline{x}) : (\mu,\nu), \Omega \mapsto \varphi$. We consider similarly all the rules in BAL.

We are ready to state completeness of the sequent calculus for BAL as a corollary from the Soundness and Completeness Theorem for MSL [23].

Corollary 2. *For every formula φ in BAL, if $\models_{\mathrm{BAL}} \varphi$, then $\vdash_{\mathrm{BAL}} \varphi$.*

Proof. Follows from Corollary 1, Theorem 2, and Soundness and Completeness Theorem for MSL, the latter theorem is proven in [23].

The results we have described in this section can be obtained, with minor modifications, for most of lattice and bilattice based annotated languages, such as [13, 15, 16, 22, 24, 25].
The many-sorted logic MSL* we defined here is in fact just a fragment of a very general many-sorted logic MSL, [23]. It is curious that the structure of MSL* is similar to the structure of the fragment of MSL which gave a translation for a second-order logic in [23]. This shows that annotated first-order many-valued logics can be equivalently represented by conventional second-order logic with sorts.
Furthermore, the way of introducing higher-order relations I in MSL* is similar to the way how [23] introduced first-order relations when translating propositional dynamic logic into many-sorted logic. It is likely that a many-sorted representation of a multimodal logic of [2], for example, will bring into the light a close connection between annotated many-valued and multimodal logics.

5 Conclusions and Further Work

We have built a sequent calculus for a very general annotated logic BAL. We used this generality to show, using the example of BAL, that annotated many-valued logics can be syntactically, semantically, and deductively translated into conventional many-sorted logic in the style of Manzano [23]. The resulting many-sorted sequent calculus has a simpler and clearer rule representation and works within conventional many-sorted language with no semantical annotations, and hence in the future may yield some conventional structural (e.g., categorical) analysis.
The uniform framework of MSL allowed us to compare properties of BAL with other non-classical logics (second order, dynamic) which have been translated into many-sorted logics already. In the future, it may be fruitful to find a many-sorted representation of a multimodal logic of [2] and show its relations with many-valued annotated logics. This would link nicely modal and many-valued logics.
Some work has been done on practical implementation of many-sorted translation to Bilattice-based Annotated Logic Programs (BAPs), see [17]. The translation made in [17] helped to simplify certain resolution rules for BAL. A rigorous analysis of efficiency of BAPs comparing with their many-sorted analogues is to be done in the future.
The further work may include the similar analysis of other many-valued annotated logics, such as logics of [7, 13, 15, 16, 21, 22, 24, 25], and some other lattice or bilattice based logics. This may lead to establishing a nice uniform framework for analysing different annotated lattice and bilattice based logics, their model and deductive properties.

References

1. Matthias Baaz, Christian G. Fremuller, and Gernot Sazler. Automated deduction for many-valued logics. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 2, pages 1355 – 1402. Elsevier, 2001.
2. M. Baldoni. *Normal Multimodal Logics: Automatic Deduction and Logic Programming extension.* PhD thesis, Torino, Italy, 2003.
3. Alonzo Church. *Introduction to Mathematical Logic.* Princeton, 1944.
4. H.D. Ebbinghaus, J.Flum, and W.Thomas. *Mathematical Logic.* Springer-Verlag, Berlin, 1984.
5. Melvin Fitting. Bilattices in logic programming. In G. Epstein, editor, *The twentieth International Symposium on Multiple-Valued Logic*, pages 238–246. IEEE, 1990.
6. Melvin Fitting. Bilattices and the semantics of logic programming. *Journal of logic programming*, 11:91–116, 1991.
7. Melvin Fitting. Many-valued modal logics. *Fundamenta informaticae*, 15:235–234, 1992.
8. Melvin Fitting. Kleene's three-valued logics and their children. *Fundamenta informaticae*, 20:113–131, 1994.
9. Melvin Fitting. Tableaus for many-valued modal logic. *Studia Logica*, 55:63–87, 1995.

10. Melvin Fitting. Bilattices are nice things. *Self-Reference*, pages 53–77, 2006.
11. Mathew L. Ginsberg. Multivalued logics: a uniform approach to reasoning in artificial intelligence. *Computational Intelligence*, 4:265–316, 1988.
12. George Grätzer. *General Lattice Theory.* Birkauser Verlag, Basel, Switzerland, 1978.
13. Reiner Hähnle. Commodious axiomatizations of quantifiers in multiple-valued logic. *Studia Logica*, 61(1):101–121, 1998.
14. Reiner Hähnle and Ganzalo Escalado-Imaz. Deduction in many-valued logics: a survey. *Mathware and soft computing*, IV(2):69–97, 1997.
15. Michael Kifer and Eliezer L. Lozinskii. RI: A logic for reasoning with inconsistency. In *Proceedings of the 4th IEEE Symposium on Logic in Computer Science (LICS)*, pages 253–262, Asilomar, 1989. IEEE Computer Press.
16. Michael Kifer and V. S. Subrahmanian. Theory of generalized annotated logic programming and its applications. *Journal of logic programming*, 12:335–367, 1991.
17. Ekaterina Komendantskaya. A many-sorted semantics for many-valued annotated logic programs. In *Proceedings of the Fourth Irish Conference on the Mathematical Foundations of Computer Science and Information Technology (MFCSIT)*, pages 225–229, Cork, Ireland, August 1– August 5 2006.
18. Ekaterina Komendantskaya and John Power. Fibrational semantics for many-valued logic programs, 2007. Submitted.
19. Ekaterina Komendantskaya, Anthony Karel Seda, and Vladimir Komendantsky. On approximation of the semantic operators determined by bilattice-based logic programs. In *Proceedings of the Seventh International Workshop on First-Order Theorem Proving (FTP'05)*, pages 112–130, Koblenz, Germany, September 15–17 2005.
20. James J. Lu. Logic programming with signs and annotations. *Journal of Logic and Computation*, 6(6):755–778, 1996.
21. James J. Lu, Neil V. Murray, and Erik Rosenthal. A framework for automated reasoning in multiple-valued logics. *Journal of Automated Reasoning*, 21(1):39–67, 1998.
22. James J. Lu, Neil V. Murray, and Erik Rosenthal. Deduction and search strategies for regular multiple-valued logics. *Journal of Multiple-valued logic and soft computing*, 11:375–406, 2005.
23. Maria Manzano. Introduction to many-sorted logic. In K. Meinke and J. V. Tucker, editors, *Many-Sorted logic and its Applications*, pages 3–88. John Wiley and Sons, UK, 1993.
24. Gernot Salzer. MUltlog 1.0: Towards an expert system for many-valued logics. In *Proc. 13th Int. Conf. on Automated Deduction (CADE'96)*, volume 1104 of *LNCS (LNAI)*, pages 226 – 230. Springer, 1996.
25. Gernot Sazler. Optimal axiomatizations of finitely-valued logics. *Information and Computation*, 162(1 – 2):185 – 205, 2000.