BSML and Left-to-right Interpretation of Natural Language

Reinhard Muskens

Institute for Logic, Language and Computation

21 March 2023

Aloni's BSML

- ⋆ Aloni (2022) introduces a bilateral state-based modal logic (BSML) as a vehicle to study free choice phenomena in natural language.
- ⋆ BSML contains what it says on the tin:
  - The logic is bilateral, with acceptance and rejection conditions for each construct.
  - The logic contains modal operators, but
  - these are not evaluated just on possible worlds, but on sets of them—states.
- ⋆ Crucial elements in Aloni's analysis of free choice phenomena are her use of bilateral negation and the split (tensor) disjunction of Cresswell (2004) and Väänänen (2007). The latter (but not the former) was also used in Hawke and Steinert-Threlkeld (2018)'s treatment of free choice.

Additional Support for Aloni's View?

- ⋆ If Aloni's analysis is right, her insight has far-reaching implications for the logical analysis of natural language. For example
  - disjunction is not what we thought it was, and
  - the ontology underlying natural language is one of states, not just of worlds or situations.
- ⋆ I will argue that there is additional support for her view that comes from considerations about semantic and pragmatic processing.
- ⋆ In particular, I will argue that a semantics based on bilateral negation and split disjunction makes it possible to model the idea that the order in which expressions are evaluated by and large is the order in which they are produced. Evaluation follows the linguistic precedence order—"from left to right" given our writing system.

Moving to Classical Type Logic

- ⋆ Natural language semantics is best studied with the help of a logic in which expressions are typed and that comes with λ-abstraction and application.
- ⋆ The classical type logic TY_2 (Church, 1940; Gallin, 1975) offers such an environment and will be the logic I will work with.
- ⋆ But BSML can be embedded within the *stt* domain of this logic, essentially by transcribing its semantic clauses.
- ⋆ While we won't formally have a bilateral state-based modal logic we will still have the possibility of a bilateral state-based modal semantics for natural language expressions.
- ⋆ In order to make this work I must first explain how to transcribe logics with a bilateral semantics in the TY_2 setting and I also need to say a few words about populating domains D_s with state-like entities.

Paired Meanings

- ⋆ A bilateral semantics gives verification and falsification (acceptance/rejection) conditions to each logical sentence. The meaning of a sentence in bilateral set-ups can therefore be characterised as a pair of classical meanings (see also Cooper, 1983).
- ⋆ Define $\star$ as $\lambda\theta''_t\theta'_t\theta_t.(\theta \to \theta'') \wedge (\neg\theta \to \theta')$ and write $\varphi \star \psi$ for $\star\varphi\psi$. Then the *tt* term $\varphi \star \psi$ is equivalent with $\lambda\theta.(\theta \to \varphi) \wedge (\neg\theta \to \psi)$ and we have that $(\varphi \star \psi)\top \equiv \varphi$, while $(\varphi \star \psi)\bot \equiv \psi$.
- ⋆ So we can form pairs $\varphi \star \psi$ in type *tt* and also have projections to retrieve their components of type *t*. An alternative would be to have a type logic with arbitrary pairing and projection.

Transcribing Bilateral Logics

- We can now emulate the operators of the Mother of All Bilateral Logics, FDE (see Anderson and Belnap (1975), Dunn (1976), Belnap (1976, 1977)). Here are its negation, conjunction, and disjunction:
  - $\lambda Z_{tt}.Z\bot \star Z\top$
  - $\lambda Z_1 Z_2.(Z_1\top \wedge Z_2\top) \star (Z_1\bot \vee Z_2\bot)$
  - $\lambda Z_1 Z_2.(Z_1\top \vee Z_2\top) \star (Z_1\bot \wedge Z_2\bot)$
- Start with expressions $p \star q$, where p and q are distinct constants of type t, close off under the operations above, then FDE entailment will correspond to $\Phi\top \models \Psi\top$ and also to $\Psi\bot \models \Phi\bot$, if Φ and Ψ belong to the sublanguage of tt expressions generated.
- Other bilateral logics can be emulated in similar ways, but for BSML we also obviously need states.

States

- ⋆ There are at least two ways to have states in our type logic.
- ⋆ One easy way is to assume a primitive type ω of worlds and to let the type s of states be ωt.
- ⋆ Another way is to impose axioms that require D_s to be an atomic boolean algebra. We then also typically want this algebra to be definably complete—every definable set of states has a join. The atoms of the algebra will function as worlds.
- ⋆ The second method will be chosen here. It allows avoiding unnecessary quantification over functions (s is now a basic type) and also makes it easier to adapt (weaken) the axioms if doing so should turn out to fit the facts in a better way.

Axioms and Definitions for States

ABA1	$\forall ij(i \geq j \wedge j \geq i \rightarrow i = j)$	(Antisymm.)
ABA2	$\forall i\ (i \geq 0 \wedge 1 \geq i) \wedge 0 \not\geq 1$	(Zero/One)
ABA3	$\forall w(Ww \leftrightarrow (0 \not\geq w \wedge \forall i(w \geq i \rightarrow i \geq w \vee 0 \geq i)))$	(Atoms)
ABA4	$\forall ij(i \geq j \leftrightarrow \forall w(Ww \wedge j \geq w \rightarrow i \geq w))$	(Inclusion)
ABA5	$\forall \vec{u} \exists k \forall w(Ww \rightarrow (k \geq w \leftrightarrow \varphi))$	(Suprema)
ABA6	$\forall ijw(Ww \rightarrow (i - j \geq w \leftrightarrow i \geq w \wedge j \not\geq w))$	(Difference)
ABA7	$\forall ijw(Ww \rightarrow (i + j \geq w \leftrightarrow i \geq w \vee j \geq w))$	(Sum)
ABA8	$\forall ijw(Ww \rightarrow (i \times j \geq w \leftrightarrow i \geq w \wedge j \geq w))$	(Product)

Axioms and definitions for definably complete atomic Boolean algebras. In ABA5 the variable k may not be free in φ.

Note that $\geq$ corresponds to $\supseteq$ in the set theoretic approach, $+$ to $\cup$, $\times$ to $\cap$, and $-$ to $\setminus$.

I'll write $\lambda w.\varphi$ for $\lambda w.Ww \wedge \varphi$, $\forall w\varphi$ for $\forall w(Ww \rightarrow \varphi)$ etc.

Transcribing BSML

- Now that we have pairing and projection, but also states, the BSML operators can be emulated. Here are negation, conjunction, disjunction, and $\Diamond$ (the Z are of type *stt*, i and j of type s, read $R[w]$ for $\Sigma w' Rww'$):
  - $\lambda Zi.Zi\bot \star Zi\top$
  - $\lambda Z_1 Z_2 i.(Z_1 i\top \wedge Z_2 i\top) \star \exists j_1 j_2(i = j_1 + j_2 \wedge Z_1 j_1\bot \wedge Z_2 j_2\bot)$
  - $\lambda Z_1 Z_2 i.\exists j_1 j_2(i = j_1 + j_2 \wedge Z_1 j_1\top \wedge Z_2 j_2\top) \star (Z_1 i\bot \wedge Z_2 i\bot)$
  - $\lambda Zi.\forall w(i \geq w \rightarrow \exists j(j \neq 0 \wedge \Sigma w' Rww' \geq j \wedge Zj\top))$
    $\star \forall w(i \geq w \rightarrow Z(\Sigma w' Rww')\bot)$
- This time, in order to get a BSML-like fragment, start with expressions of the form $\lambda i.\forall w(i \geq w \rightarrow pw) \star \forall w(i \geq w \rightarrow \neg pw)$, where p is a constant of type *st*, add $\lambda i.i \neq 0 \star i = 0$ (i.e. NE), and close off under the operations above. BSML entailment should correspond to $\Phi \mathtt{i}\top \models \Psi \mathtt{i}\top$ ($\mathtt{i}$ an arbitrary constant of type s), if Φ and Ψ belong to the sublanguage of *stt* expressions generated.

What I Will Actually Use 1

The following are meaning postulate schemes for constants `not`, `or`, and `and`. Any universal closure of an instantiation of the metavariables p, q, and i is a meaning postulate.

(1) a. $\mathtt{not}\ \mathtt{pi}\top \leftrightarrow \mathtt{pi}\bot$
    b. $\mathtt{not}\ \mathtt{pi}\bot \leftrightarrow \mathtt{pi}\top$

(2) a. $\mathtt{or}\ \mathtt{pqi}\top \leftrightarrow \exists jj'(j' + j = \mathtt{i} \wedge \mathtt{p}j'\top \wedge \mathtt{p}j\bot \wedge \mathtt{q}j\top)$
    b. $\mathtt{or}\ \mathtt{pqi}\bot \leftrightarrow \mathtt{pi}\bot \wedge \mathtt{qi}\bot$

(3) a. $\mathtt{and}\ \mathtt{pqi}\top \leftrightarrow \mathtt{pi}\top \wedge \mathtt{qi}\top$
    b. $\mathtt{and}\ \mathtt{pqi}\bot \leftrightarrow \exists jj'(j' + j = \mathtt{i} \wedge \mathtt{p}j'\bot \wedge \mathtt{p}j\top \wedge \mathtt{q}j\bot)$

Note that (2a) and (3b) are strengthened here (but if it is assumed that p and q are flat these clauses are still equivalent with the original ones).

What I Will Actually Use 2

Here is a preliminary meaning postulate scheme for $\texttt{may}$. Owj is short for $\forall w'(j \geq w' \rightarrow Oww')$, where O is the deontic accessibility relation.

(4) (to be revised)

a. $\texttt{may}\ \texttt{p}\texttt{i}\top \leftrightarrow \forall w(\texttt{i} \geq w \rightarrow \exists j(j \neq 0 \wedge Owj \wedge \texttt{p}j\top))$

b. $\texttt{may}\ \texttt{p}\texttt{i}\bot \leftrightarrow \forall w(\texttt{i} \geq w \rightarrow \forall j(Owj \rightarrow \texttt{p}j\bot))$

Note that the treatment in terms of $O[w]$ has been replaced by one that is a bit more direct.

Left-to-right Evaluation and Presupposition

- ⋆ Let's get back to my motivation for stealing ideas from BSML: left-to-right evaluation of natural language. Why is it important?
- ⋆ Answer: because evaluation has side-effects. Evaluation of an item always is relative to context but also updates that context. A next item is then evaluated relative to the increased context.
- ⋆ The semantic presuppositions of an item must be entailed by the context (the pragmatic presupposition) in which the item is evaluated.
- ⋆ John has a sister and he will drive his sister to the airport.
- ⋆ The central role of left-to-right processing and its influence on the local context and the satisfaction conditions of semantic presuppositions was already stressed in Stalnaker's and Karttunen's pivotal early work (Stalnaker 1973; Karttunen 1973, 1974), and more recently has again been emphasised by Schlenker (2009, 2010) and Barker (2022).

Presupposition and Abduction

- ⋆ But left-to-right evaluation cannot be the whole story behind presuppositional phenomena. A second piece of the puzzle that seems essential is accommodation (Lewis, 1979).
- ⋆ Roughly, if the context in which an item is evaluated does not entail a semantic presupposition triggered by that item the context will, if possible, be enriched before that evaluation takes place, in such a way that the enriched context does entail the presupposition.
- ⋆ But the enrichment need not be minimal. It may be ampliative and in fact accommodation bears all the hallmarks of abduction.

Interpretation as Abduction

- ⋆ You observe that the pavement is wet (C). You know that whenever it has rained the pavement is wet (P1). In order to explain C, you abductively infer that it has rained (P2).
- ⋆ Hobbs et al. (1993): "[...] to interpret a text, one must prove the logical form of the text from what is already mutually known, [...] making assumptions where necessary."
- ⋆ This is an important insight that explains a lot about the interaction of "given" and "new" information.
- ⋆ I will build upon Hobbs's insight, but will deviate technically from his approach. In particular, I'll use tableau abduction (see Mayer and Pirri 1993; Aliseda-Llera 1997; D'Agostino et al. 2008; Kohlhase 1995) to formalise things.

A Tableau Calculus I

- ⋆ On a following slide the rules of a certain tableau calculus for first-order logic are presented.
- ⋆ Tableaux will be ordered. They can be drawn as trees, in the usual way, but, formally, branches will be lists (finite sequences) of formulas and tableaux will be lists of branches.
- ⋆ The basic idea here is that an order of evaluation is imposed that must essentially follow the linguistic precedence order.

A Tableau Calculus II

- ⋆ The tableau calculus is not intended to characterise first-order entailment and it doesn't. It's intended to enable abduction.
- ⋆ The expansion rules for propositional connectives will be standard ones (modulo the imposition of order).
- ⋆ The expansion rules for the quantifiers make tableaux into a free variable tableaux, but they are dual to the ones in the usual free variable tableaux.
- ⋆ Repetition of rule applications is not allowed and there is no general rule that allows for applying substitutions to tableaux (as in standard free variable calculi).
- ⋆ I'll present the rules first and then give a characterisation of what they do.

Tableau Rules

$$\begin{array}{c}\varphi \land \psi\\ |\\ \varphi\\ \psi\end{array} \qquad \begin{array}{cc}\multicolumn{2}{c}{\neg(\varphi \land \psi)}\\ / & \backslash\\ \neg\varphi & \neg\psi\end{array} \qquad \begin{array}{cc}\multicolumn{2}{c}{\varphi \lor \psi}\\ / & \backslash\\ \varphi & \psi\end{array} \qquad \begin{array}{c}\neg(\varphi \lor \psi)\\ |\\ \neg\varphi\\ \neg\psi\end{array}$$

$$\begin{array}{cc}\multicolumn{2}{c}{\varphi \to \psi}\\ / & \backslash\\ \neg\varphi & \psi\end{array} \qquad \begin{array}{c}\neg(\varphi \to \psi)\\ |\\ \varphi\\ \neg\psi\end{array} \qquad \begin{array}{c}\neg\neg\varphi\\ |\\ \varphi\end{array}$$

$$\begin{array}{c}\exists u\, \varphi\\ |\\ \varphi\{u := v\}\end{array} \qquad \begin{array}{c}\forall u\, \varphi\\ |\\ \varphi\{u := \gamma\vec{v}\}\end{array} \qquad \begin{array}{c}\neg\exists u\, \varphi\\ |\\ \neg\varphi\{u := \gamma\vec{v}\}\end{array} \qquad \begin{array}{c}\neg\forall u\, \varphi\\ |\\ \neg\varphi\{u := v\}\end{array}$$

Side Conditions, Closure, and Characterising Property

- ⋆ The v that is created in the rule for $\exists$ must be fresh to the branch and will be called an independent variable.
- ⋆ The Skolem variable (or dependent variable) γ in the rule for $\forall$ must be fresh to the tree. The $\vec{v}$ in that rule must be the independent variables that were created on the branch that far (in the order they were created).
- ⋆ Rule applications erase their input formula.
- ⋆ A branch that contains literals α and $\neg\alpha$ is closed.
- ⋆ Branches can also be closed by the context, as will be seen shortly.
- ⋆ Let $\mathcal{T}$ be a completed tableau for φ, let $\mathcal{B}_1, \ldots, \mathcal{B}_n$ be the branches of $\mathcal{T}$, and let, for each i, L_i be the conjunction of literals on $\mathcal{B}_i$. Then φ is equivalent with $\forall\vec{\gamma}\exists\vec{v}(L_1 \vee \ldots \vee L_n)$, where the $\vec{\gamma}$ are the Skolem variables and the $\vec{v}$ the independent variables in $\mathcal{T}$.

Abductive Inference

- ⋆ In order to abductively infer φ from context (and increase the context as a result), develop a tableau for $\neg\varphi$.
- ⋆ Some branches may close, for example if they conflict with a clause in the context.
- ⋆ Each remaining open branch of the tableau corresponds with a way φ may be false in the context. (It's not guaranteed. If it isn't, a redundant update of context will follow.)
- ⋆ For each such open branch a clause must be added to the context to close it. We generalise the branch-driven abduction of D'Agostino et al. (2008).

Meaning Recipes

We will start with a language of meaning recipes—λ-terms that encode how the meaning of an expression is composed from the meaning of its parts (Benthem, 1988, 1991). The following are examples of meaning recipes of type *stt*. They are very close to syntactic representations.

(5) a. Every man loves a woman
  b. $((\texttt{a woman})(\lambda y.((\texttt{every man})(\lambda x.((\texttt{love}\ y)x)))))$
  c. $((\texttt{every man})(\lambda x.((\texttt{a woman})(\lambda y.((\texttt{love}\ y)x)))))$

(6) a. The king is not bald
  b. $(\texttt{not}((\texttt{the}_1\ \texttt{king})(\texttt{is bald})))$
  c. $((\texttt{the}_1\ \texttt{king})(\lambda x.(\texttt{not}((\texttt{is bald})x))))$

(7) a. Ann believes that Mary knows that the king is not bald
  b. $(\texttt{ann}(\texttt{believe}(\texttt{mary}(\texttt{know}(\texttt{not}((\texttt{the}_1\ \texttt{king})(\texttt{is bald})))))))$

The constants `not`, `and`, `or`, and `may` we have seen before are examples of ("abstract") constants that may occur in meaning recipes.

From Meaning Recipes to Clauses

- ⋆ In order to interpret a meaning recipe like (5c), apply it to a constant `@` of type s (the current state) and $\top$ (polarity is positive) and negate the result, so that, say, $\neg$(5c)`@`$\top$ is obtained.
- ⋆ Each abstract constant comes with two meaning postulate schemes (we have seen those for `not`, `and`, `or`, and `may`). These can be "compiled out" to obtain derived rules, two for each constant, as will be illustrated on the next slides.
- ⋆ With the help of these develop a tableau (in this case for $\neg$(5c)`@`$\top$) in left-to-right order.
- ⋆ From left to right, contradict the "concrete" literals on each branch with the help of a suitable clause (there is always a minimal one—the "Least Compromising Hypothesis" of D'Agostino et al. 2008).

Rules for king, love and not

$$\begin{array}{c} \neg\mathtt{king}\ \mathtt{ti}\top \\ | \\ \mathtt{i} \geq w \\ \neg \mathit{king}\,\mathtt{t}w \end{array} \qquad \begin{array}{c} \neg\mathtt{king}\ \mathtt{ti}\bot \\ | \\ \mathtt{i} \geq w \\ \mathit{king}\,\mathtt{t}w \end{array}$$

$$\begin{array}{c} \neg\mathtt{love}\ \mathtt{t'ti}\top \\ | \\ \mathtt{i} \geq w \\ \neg \mathit{love}\,\mathtt{tt'}w \end{array} \qquad \begin{array}{c} \neg\mathtt{love}\ \mathtt{t'ti}\bot \\ | \\ \mathtt{i} \geq w \\ \mathit{love}\,\mathtt{tt'}w \end{array}$$

$$\begin{array}{c} \neg\mathtt{not}\ \mathtt{pi}\top \\ | \\ \neg\mathtt{pi}\bot \end{array} \qquad \begin{array}{c} \neg\mathtt{not}\ \mathtt{pi}\bot \\ | \\ \neg\mathtt{pi}\top \end{array}$$

Rules for or, and, and if

$$\neg\mathtt{or}\ \mathtt{pqi}\top$$

$$s\vec{v}+s'\vec{v}\neq \mathtt{i} \qquad \neg\mathtt{p}(s'\vec{v})\top \qquad \neg\mathtt{p}(s\vec{v})\bot \qquad \neg\mathtt{q}(s\vec{v})\top$$

$$\neg\mathtt{or}\ \mathtt{pqi}\bot$$

$$\neg\mathtt{pi}\bot \qquad \neg\mathtt{qi}\bot$$

$$\neg\mathtt{and}\ \mathtt{pqi}\top$$

$$\neg\mathtt{pi}\top \qquad \neg\mathtt{qi}\top$$

$$\neg\mathtt{and}\ \mathtt{pqi}\bot$$

$$s\vec{v}+s'\vec{v}\neq \mathtt{i} \qquad \neg\mathtt{p}(s'\vec{v})\bot \qquad \neg\mathtt{p}(s\vec{v})\top \qquad \neg\mathtt{q}(s\vec{v})\bot$$

$$\neg\mathtt{if}\ \mathtt{pqi}\top$$

$$s\vec{v}+s'\vec{v}\neq \mathtt{i} \qquad \neg\mathtt{p}(s'\vec{v})\bot \qquad \neg\mathtt{p}(s\vec{v})\top \qquad \neg\mathtt{q}(s\vec{v})\bot$$

$$\neg\mathtt{if}\ \mathtt{pqi}\bot$$

$$\neg\mathtt{pi}\top \qquad \neg\mathtt{qi}\bot$$

Note how these rules lead to left-to-right evaluation!

Revised Postulates for may

- Deontic may seems to carry a presupposition of indisputability (for this notion, see Aloni 2022).
- Presuppositions can be treated as conditions on meaning postulates. Here are the revised postulates for may:
  - $\forall w_1 w_2 w_3(\mathtt{i} \geq w_1 \wedge \mathtt{i} \geq w_2 \wedge Ow_1 w_3 \rightarrow Ow_2 w_3) \rightarrow$
    $(\mathtt{may}\ \mathtt{pi}\top \leftrightarrow \forall w(\mathtt{i} \geq w \rightarrow \exists j(j \neq 0 \wedge Owj \wedge \mathtt{p}j\top)))$
  - $\forall w_1 w_2 w_3(\mathtt{i} \geq w_1 \wedge \mathtt{i} \geq w_2 \wedge Ow_1 w_3 \rightarrow Ow_2 w_3) \rightarrow$
    $(\mathtt{may}\ \mathtt{pi}\bot \leftrightarrow \forall w(\mathtt{i} \geq w \rightarrow \forall j(Owj \rightarrow \mathtt{p}j\bot)))$
- The following slide will show the resulting derived tableau rules for deontic may.

Rules for may

$$
\begin{array}{c}
\neg\mathtt{may}\ \mathtt{pi}\top \\
\begin{array}{cc}
\begin{array}{c}
\pi \\
| \\
\mathtt{i} \geq w_1 \\
\mathtt{i} \geq w_2 \\
O w_1 w_3 \\
\neg O w_2 w_3
\end{array}
&
\begin{array}{c}
\mathtt{i} \geq w \\
\begin{array}{ccc}
s\vec{v} = 0 & \neg O w(s\vec{v}) & \neg\mathtt{p}(s\vec{v})\top
\end{array}
\end{array}
\end{array}
\end{array}
\qquad
\begin{array}{c}
\neg\mathtt{may}\ \mathtt{pi}\bot \\
\begin{array}{cc}
\begin{array}{c}
\pi \\
| \\
\mathtt{i} \geq w_1 \\
\mathtt{i} \geq w_2 \\
O w_1 w_3 \\
\neg O w_2 w_3
\end{array}
&
\begin{array}{c}
\mathtt{i} \geq w \\
O w j \\
\neg\mathtt{p} j \bot
\end{array}
\end{array}
\end{array}
$$

(The π entries are not formulas, but markers saying that the material below them is presupposed.)

Rules for every

$$
\begin{array}{c}
\neg\texttt{every PP}'\texttt{i}\top \\
| \\
\texttt{i} \geq w \\
Exw \\
\begin{array}{cccc}
s\vec{v} + s'\vec{v} \neq \texttt{i} & \neg\mathrm{P}x(s'\vec{v})\bot & \neg\mathrm{P}x(s\vec{v})\top & \neg\mathrm{P}'x(s\vec{v})\top
\end{array}
\end{array}
$$

$$
\begin{array}{c}
\neg\texttt{every PP}'\texttt{i}\bot \\
| \\
\texttt{i} \geq w \\
\begin{array}{ccc}
\neg E(f\vec{v})w & \neg\mathrm{P}(f\vec{v})w\top & \neg\mathrm{P}'(f\vec{v})w\bot
\end{array}
\end{array}
$$

Some Properties

- ⋆ Trees starting with a formula $\neg\mathtt{MR}@\top$, where $\mathtt{MR}$ is a meaning recipe have some special properties.
- ⋆ Each branch contains at most one formula that is complex, i.e. is not a concrete literal (remember that rules erase their input). [The underlying bilateral state-based semantics makes it the case that at no point a classical disjunction of complex formulas is encountered.]
- ⋆ The active formula of a non-finished tableau is the unique complex formula on the first branch that contains one. This is the only formula that can be rewritten.
- ⋆ Complex formulas always start with $\neg$.
- ⋆ At any stage, the order of complex formulas in the tree reflects a "default" word order. (We may want to give additional rules for marked word orders.)
- ⋆ The notion of local context becomes easily definable.

Conclusion

- ⋆ There are reasons to believe that natural language interpretation by and large follows the order of words.
- ⋆ Presuppositional phenomena point in this direction, but they also seem to support the idea that interpretation is abductive.
- ⋆ A third thing that seems to be the case is that interpretation is compositional.
- ⋆ In this talk I have sketched a model that gives an account of this. A meaning recipe that encodes how compositional interpretation should take place is found and interpretation in context is then modelled by an abductive tableau procedure that works in tandem with a pragmatic process that closes tree branches and updates context.
- ⋆ Branches of the interpretation tree are ordered in a way that follows word order. Modelling things in this way makes essential use of the bilateral state-based approach and split disjunction.

Thank You!

Questions?

References I

A. Aliseda-Llera. *Seeking Explanations: Abduction in Logic, Philosophy of Science and Artificial Intelligence*. PhD thesis, ILLC, 1997.

Maria Aloni. Logic and conversation: The case of free choice. *Semantics and Pragmatics*, 15(5), 2022. Early access.

A. R. Anderson and N. D. Jr. Belnap. *Entailment: the Logic of Relevance and Necessity, Vol I*. Princeton University Press, Princeton, 1975.

Chris Barker. Composing local contexts. *Journal of Semantics*, 39(2): 385–407, 2022.

N. D. Belnap. How a Computer Should Think. In G. Ryle, editor, *Contemporary Aspects of Philosophy*, pages 30–56. Oriel Press, Stocksfield, 1976.

N. D. Belnap. A Useful Four-Valued logic. In J. M. Dunn and G. Epstein, editors, *Modern Uses of Multiple-Valued Logic*, pages 8–37. Reidel, Dordrecht, 1977.

References II

J. F. A. K. van Benthem. The Semantics of Variety in Categorial Grammar. In W. Buszkowski, W. Marciszewski, and J. F. A. K. van Benthem, editors, *Categorial Grammar*, pages 37–55. John Benjamins, Amsterdam, 1988.

J. F. A. K. van Benthem. *Language in Action*. North-Holland, Amsterdam, 1991.

A. Church. A Formulation of the Simple Theory of Types. *Journal of Symbolic Logic*, 5:56–68, 1940.

R. Cooper. *Quantification and Syntactic Theory*. Reidel, Dordrecht, 1983.

Max Cresswell. Possibility semantics for modal logic. *Australasian Journal of Logic*, 2:11–29, 2004.

M. D'Agostino, M. Finger, and D. Gabbay. Cut-Based Abduction. *Logic Journal of the IGPL*, 16(6):537–560, 2008.

References III

J. M. Dunn. Intuitive Semantics for First-Degree Entailments and 'Coupled Trees'. *Philosophical Studies*, 29:149–168, 1976.

D. Gallin. *Intensional and Higher-Order Modal Logic*. North-Holland, Amsterdam, 1975.

Peter Hawke and Shane Steinert-Threlkeld. Informational dynamics of epistemic possibility modals. *Synthese*, 195(10):4309–4342, 2018.

J. Hobbs, M. Stickel, D. Appelt, and P. Martin. Interpretation as Abduction. *Artificial Intelligence*, 63(1–2):69–142, 1993.

L. Karttunen. Presupposition and Linguistic Context. *Theoretical Linguistics*, 1:181–194, 1974.

Lauri Karttunen. Presuppositions of compound sentences. *Linguistic Inquiry*, 4(2):181–194, 1973.

References IV

Michael Kohlhase. Higher-order tableaux. In Peter Baumgartner, Reiner Hähnle, and Joachim Possega, editors, *Theorem Proving with Analytic Tableaux and Related Methods*, pages 294–309, Berlin, Heidelberg, 1995. Springer.

D. Lewis. Scorekeeping in a Language Game. *Journal of Philosophical Logic*, 8:339–359, 1979.

M. Cialdea Mayer and F. Pirri. First Order Abduction via Tableau and Sequent Calculi. *Bulletin of the IGPL*, 1(1):99–117, 1993.

Philippe Schlenker. Local contexts. *Semantics & Pragmatics*, 2:1–78, 2009.

Philippe Schlenker. Local contexts and local meanings. *Philosophical Studies*, 151:115–142, 2010.

Robert Stalnaker. Presuppositions. *Journal of Philosophical Logic*, 2(4): 447–457, 1973.

References V

Jouko Väänänen. *Dependence Logic: A New Approach to Independence Friendly Logic*. Cambridge University Press, Cambridge, 2007.