

Formalizing AMR Inference via Hybrid Logic Tableaux

CL Masters Thesis Defense

Eli Goldner

August 10, 2021

Introduction

Semantic representation:

- ▶ Capture meaning of natural language content.
- ▶ Designed for software to manipulate and interpret.

Abstract Meaning Representation (AMR):

- ▶ Graph-based (DAG), nodes are *concepts*, edges are *relations*.
- ▶ Built on predicative core of a sentence.
- ▶ Ignores syntactic and fine grained semantic differences between sentences.

Introduction

(Core) AMR is reductionistic. Optimal for:

- ▶ Annotation (esp. by non-experts).
- ▶ Semantic parsing (smaller target space).

Not optimal for:

- ▶ Representing and recovering fine-grained meaning.
- ▶ Automating reasoning/inference.

Introduction

- ▶ AMR designers made a choice that works well in data-driven NLP.
- ▶ Modular AMR extensions.
- ▶ Some extensions afford richer logical interpretation:

Motivation

"Why do we need formal methods? Can't state-of-the-art language models do this already?"

- ▶ Statistically driven techniques are unnecessarily expensive for formal inference.
- ▶ Increasing need for ability to guarantee/verify properties of software:
 - ▶ Does the software give us the right *type* of result for an input?
 - ▶ Bias in NLP.

Approach

- ▶ Combine two AMR extensions for richer interpretation:
 - ▶ Scope and quantification (Pustejovsky et al., 2019)
 - ▶ Tense and aspect (Donatelli et al., 2018)
- ▶ Interpret these extended AMR into a logic that handles quantification and tense.
- ▶ Develop tableau methods for this logic:
 - ▶ General method for proving/disproving sentences in the logic.
 - ▶ Restricted method for checking if sentence holds in some model.

AMR with Scope and Quantification

- ▶ Disambiguates scope.
- ▶ Annotates central predicate and its arguments.
- ▶ Clearest path for AMR → standard first-order predicate logic.

AMR with Tense and Aspect

- ▶ Standard AMR structure.
- ▶ Central predicate annotated for:
 - ▶ Aspect.
 - ▶ Speech time.
 - ▶ Reference time.

Combined Extensions

- ▶ Assume each AMR has information from both extensions.
- ▶ Attach tense and aspect information to central predicate node.
- ▶ Extract a tense-sensitive FOPL representation (details later).

Modal Logic

- ▶ Propositional logic lets us form statements like $p \wedge (q \vee \neg r)$.
- ▶ Modal propositional logic extends propositional logic with an operator \Diamond
- ▶ It is not possible that p and $\neg p$ are the case:

$$\neg\Diamond(p \wedge \neg p)$$

- ▶ Meaning of \Diamond : There is a *possible world* where p is true, and this possible world is *accessible* from the current one.
- ▶ The problem: the “current world” is implicit and dependent on context.
- ▶ How to make it explicit?

Hybrid Logic

- ▶ Extend propositional modal logic, with the operator @ , which specifies the world we're referring to.
- ▶ p or r is possible at world i :

$$\text{@}_i \Diamond (p \vee r)$$

- ▶ In the above proposition i is called a nominal since it *names* some/is true at exactly one world.
- ▶ Everything true at the nominal j is true at the nominal i (they name the same world): $\text{@}_i j$

Hybrid Logic Variants

Hybrid tense logic:

- ▶ Two tense modalities: F and P
- ▶ World j is in the past of world i : $\text{@}_i P j$

Quantified hybrid logic:

- ▶ Hybrid logic with first-order quantifiers, relation, and function symbols.
- ▶ At n (now) there is a person for whom it is possible to own a car:

$$\text{@}_n (\exists x)(\text{Person}(x) \wedge (\exists y)(\text{Car}(y) \wedge \Diamond \text{Afford}(x, y)))$$

Hybrid Logic Variants

A problem

$$\textcircled{O}_n(\exists x)(\textit{Person}(x) \wedge (\exists y)(\textit{Car}(y) \wedge \Diamond \textit{Afford}(x, y)))$$

What's the domain of quantification?

- ▶ Option 1: The domain is the same at every world: *There is someone (out of all people all people at all worlds) who can afford some car (out of all cars at all worlds).*
- ▶ Option 2: Each world has a different domain: *There is someone (out of everyone in this world) who can afford some car (out of all cars in this world).*

Hybrid Logic Variants

- ▶ We tend to mean there is some person (and more loosely some car) here/now.
- ▶ Option 2 is closer to how quantification works in natural language.
- ▶ We use this approach (presentist quantification).

First-Order Hybrid Tense Logic

First-order Hybrid Tense Logic (*FHTL*) is:

- ▶ Quantified hybrid logic, with tense modalities F and P instead of \Diamond .
- ▶ Presentist quantification:
 - ▶ Each world has its own domain.
 - ▶ Quantification is domain sensitive.

First-Order Hybrid Tense Logic

In logic models are interpretations of symbols and constants in the language. An *FHTL* model \mathfrak{M} is a tuple

$$(T, \mathcal{R}, (D_t)_{t \in T}, I_{nom}, (I_t)_{t \in T})$$

Where:

- ▶ T is a set of times/worlds.
- ▶ \mathcal{R} is the binary accessibility relation over times.
- ▶ D_t is the domain of a time t
- ▶ I_{nom} assigns nominals to worlds.
- ▶ I_t interprets the value of terms at a time t .

First-Order Hybrid Tense Logic

- ▶ The satisfiability of a formula with free variables depends on a *variable assignment function*.
- ▶ A formula with no free variables is called a *sentence*.

Two main tasks for a sentence $\Theta_s \varphi$:

- ▶ Theorem proving – is $\Theta_s \varphi$ true regardless of the model?
- ▶ Model checking – is $\Theta_s \varphi$ true in a given model \mathfrak{M} ?

For both we use versions of the tableau method.

Tableau Method

- ▶ Tableau for a formula is a tree structure.
- ▶ A tableau calculus for a logic breaks down and transforms formulae
- ▶ Rules of the calculus use semantics of connectives/quantifiers/modifiers.
- ▶ *FHTL* tableau based on *QHL* tableau modified for tense rules.

Tableau Method

- ▶ A tableau branch is a subtree of the main tableau tree.
- ▶ A tableau *branches* for rules involving disjunctions.
- ▶ A branch is *closed* if it contains both a formula $\mathbb{O}_s\varphi$ and its negation $\mathbb{O}_s\neg\varphi$. Otherwise it is open.
- ▶ A branch is *saturated* if no rules of the calculus can be applied without adding a redundant formula on the branch.

Tableau Example Rules

$$\frac{\textcircled{O}_s F \varphi}{\textcircled{O}_s Fa} (F)^{12}$$

$\textcircled{O}_a \varphi$

$$\frac{\textcircled{O}_s \textcircled{O}_t \varphi}{\textcircled{O}_t \varphi} (\textcircled{O})$$

$$\frac{\textcircled{O}_s (\exists x) \varphi}{\textcircled{O}_s \varphi [s:p/x]} (\exists)^3$$

$$\frac{\textcircled{O}_i j:t = k:s \quad \textcircled{O}_i \varphi}{\textcircled{O}_i \varphi [j:t//k:s]} (\textbf{sub})^4$$

¹The nominal a is new to the branch.

²The formula φ is not a nominal.

³ $s:p$ is new to the branch.

⁴ $\varphi[j:t//k:s]$ is φ where some occurrences of $j:t$ have been replaced by $k:s$.

FHTL Tableau Example

- | | |
|-----|----------------------------------------------------------------------------------------------|
| (1) | |
| (2) | $\text{@}_s(\exists x)[P((\exists y)[f(x, y) = f(y, x)]) \vee \neg(\exists z)[x = z]]$ |
| (3) | $\text{@}_s P((\exists y)[f(s_1, y) = f(y, s_1)]) \vee \neg(\exists z)[s_1 = z]$ |
| | \diagup |
| (4) | $\text{@}_s P((\exists y)[f(s_1, y) = f(y, s_1)]) \quad \text{@}_s \neg(\exists z)[s_1 = z]$ |
| (5) | $\text{@}_s Pt \qquad \qquad \qquad \text{@}_s \neg[s_1 = s_1]$ |
| (6) | $\text{@}_t(\exists y)[f(s_1, y) = f(y, s_1)] \qquad \qquad \text{@}_s [s_1 = s_1]$ |
| (7) | \dots \otimes |

Tableau Proofs

- ▶ The *root formula* of a tableau is unsatisfiable if every branch of the tableau closes.
- ▶ If we want to prove $\mathbb{Q}_s\varphi$ then we begin the tableau with $\mathbb{Q}_s\neg\varphi$ (proof by contradiction).
- ▶ Need to show for tableau method (or any proof system) that it is:
 - ▶ Sound – if a tableau is closed then the root formula is unsatisfiable.
 - ▶ Complete – if a formula is unsatisfiable it has a closed tableau proof.

Tableau Proofs

- ▶ Soundness is demonstrated by checking the rules.
- ▶ Completeness is demonstrated by contrapositive.
- ▶ Open tableau branch → there is a model which satisfies the root.
- ▶ Construct the model out of equivalence classes of everything that shows up on the branch.

Tableaux for Model Checking

- ▶ So far the tableau method has been about general theorem proving.
- ▶ Problem: no tableau construction method guaranteed to terminate (if there was we'd have decidability of first-order logic).
- ▶ Most of the time however we are reasoning about an AMR/*FHTL* sentence in some local context.
- ▶ Revise tableau rules, trade completeness for termination.

Tableaux for Model Checking

NB: Terms in *FHTL* have nominal prefixes since their value can depend on the world they are evaluated in, i.e. $i:t$, $i:j:t$. There are three things that prevent tableau construction from terminating:

- ▶ Universal tableau rules ($\neg\exists$, \forall) – can generate an arbitrary number of conclusions.
- ▶ Term rules – can keep adding prefixes to terms.
- ▶ Nominal rules – generating redundant nominals (our complete rules take care of this).

- ▶ Idea: give tableau a reasonable chance to close, but make sure it terminates.
- ▶ Fix some $q \in \mathbb{N}^+$ and let universal rules generate at most q conclusions. (Fitting (1988) says $q = 1$ works surprisingly often.)
- ▶ Restrict term rules to depend on nominals and terms already on the branch. Prevent redundant prefixes (no $i:j:i:t$ etc.).

Model checking procedure

- ▶ Build tableau using restricted rules.
- ▶ Check tableau tree from leaves up, skipping closed branches.
- ▶ Every formula on the branch has a set of variable and parameter assignments which satisfy it in the model.
- ▶ Solve for a formula's set based on the sets of its conclusions.
- ▶ If the root formula is satisfied in the model by every variable assignment then it holds.

Model Checking Example

- ▶ Every desk will have a computer located there.

- ▶ AMR with quantification and tense:

(s / scope

```
:pred (b / be-located-at-91 :ongoing -
      :complete +
      :time (a / after
            :op1 (n / now))

      :ARG0 (c / computer)
      :ARG1 (d / desk
            :quant (e / every)))
```

```
:ARG0 d
:ARG1 c)
```

- ▶ FHTL translation:

$$@_{now}(\forall y)[\text{desk}(y) \rightarrow (\exists x)[\text{computer}(x) \wedge F(\text{be-located-at-91}(x, y))]]$$

Model Checking Example

Define a small *FHTL* model $\mathfrak{M} = (T, \mathcal{R}, (D_t)_{t \in T}, I_{nom}, (I_t)_{t \in T})$ where:

$$T = \{\text{yesterday, now, tomorrow}\}$$

$$\mathcal{R} = \{(\text{yesterday, now}), (\text{now, tomorrow}), (\text{yesterday, tomorrow})\}$$

$$I_{nom} = \{(\text{y, yesterday}), (\text{n, now}), (\text{t, tomorrow})\}$$

$$I_{\text{yesterday}}(\text{be-located-at-91}) = \dots$$

$$I_{\text{now}}(\text{be-located-at-91}) = \dots$$

$$I_{\text{tomorrow}}(\text{be-located-at-91}) = \{\langle \text{computer}_1, \text{desk}_2 \rangle, \langle \text{computer}_2, \text{desk}_1 \rangle\}$$

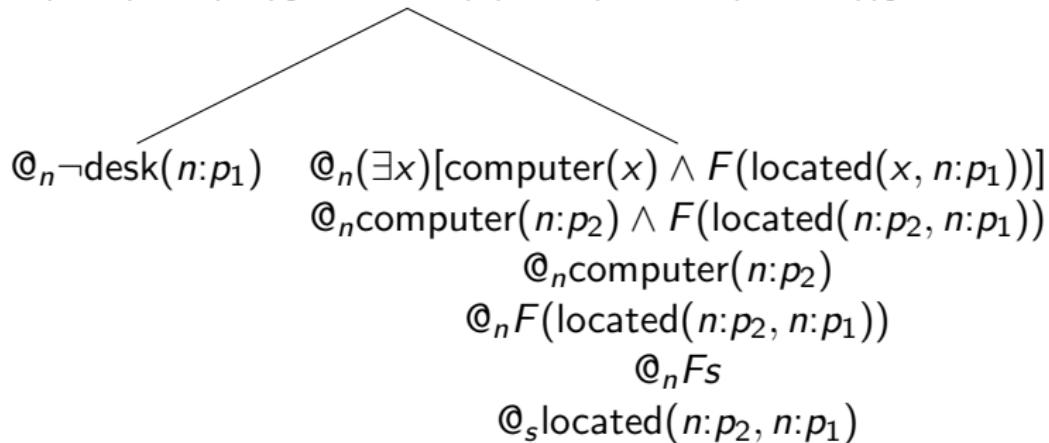
$$D_{\text{yesterday}} = \{\text{computer}_1, \text{desk}_1\}$$

$$D_{\text{now}} = \{\text{computer}_1, \text{computer}_2, \text{desk}_1, \text{desk}_2, \text{desk}_3\}$$

$$D_{\text{tomorrow}} = \{\text{computer}_1, \text{computer}_2, \text{desk}_1, \text{desk}_2\}$$

Model Checking Example

$$\begin{aligned} @_n(\forall y)[\text{desk}(y) \rightarrow (\exists x)[\text{computer}(x) \wedge F(\text{located}(x, y))]] \\ @_n \text{desk}(n:p_1) \rightarrow (\exists x)[\text{computer}(x) \wedge F(\text{located}(x, n:p_1))] \end{aligned}$$



Where we assign $s = t = \text{tomorrow}$, $n:p_2 = \text{computer}_1$ and $n:p_1 = \text{desk}_2$, or $n:p_2 = \text{computer}_2$ and $n:p_1 = \text{desk}_1$ we see that \mathfrak{M} satisfies the *FHTL* sentence.

Extraction

In this new annotation time information is of the form:

:<aspect> <polarity> :time (<reference> :op1 (<speech>))

- ▶ <speech> is some $t \in T$, so formula has the prefix $\text{@}_t\varphi$
- ▶ <reference> (if present) is the tense, e.g. $\text{@}_tP\psi$
- ▶ We don't make use of aspect (right now).

Interpretation

Every AMR in this annotation is either of the form (for a connective or, and, or cond):

(<connective>

:op1 (...)
:op2 (...)
:op3 (...))

Or

(s / scope

:pred (...
:ARG0 (...)
:ARG1 (...))
:ARG0 <primary-scope>
:ARG1 <secondary-scope>)

Interpretation Examples

```
(o / or
  :op1 (...))
  :op2 (...))
  :op3 (...))
                           [[op1]] ∨ [[op2]] ∨ [[op3]]
```

```
(s / scope
  :pred ...
    :ARG0 (...))
    :ARG1 (...))
  :ARG0 <primary-scope>
  :ARG1 <secondary-scope>)

                           [[<primary-scope>]]([[<secondary-scope>]]([[pred]]))
```

Interpretation Examples

```
(s / scope
  :pred (b / be-located-at-91 :ongoing -
          :complete +
          :time (a / after
                 :op1 (n / now)))
  :ARG0 (c / computer)
  :ARG1 (d / desk
         :quant (e / every)))
:ARG0 d
:ARG1 c)
```

$$@_{\text{now}}(\forall y)[\text{desk}(y) \rightarrow (\exists x)[\text{computer}(x) \wedge F(\text{be-located-at-91}(x, y))]]$$

- [[now]] $\triangleright now$
- [[after]] $\triangleright F$
- (c / computer)
- $$\lambda\varphi.(\exists x)[Computer(x) \wedge \varphi(x)]$$
- (d / desk
:quant (e/ every))
- $$\lambda\varphi.(\forall x)[Desk(x) \rightarrow \varphi(x)]$$

Interpretation Example

```
(:pred (b / be-located-at-91 :ongoing -  
           :complete +  
           :time (a / after  
                  :op1 (n / now))
```

$$\lambda\varphi.\lambda x.\lambda y.\varphi(\text{be-located-at-91}(x, y))$$

Interpretation Example

First application:

$\llbracket \text{computer will be located at it} \rrbracket =$

$$\lambda\varphi.(\lambda\varphi'.(\exists c)[\text{Computer}(c) \wedge \varphi'(c)])(\lambda\psi.(\lambda\psi'.\psi'(\lambda x.\lambda y.F(loc(x,y))))(\lambda\gamma.\varphi(\gamma(\psi)))) \triangleright_{\beta}$$

$$\lambda\varphi.(\exists c)[\text{Computer}(c) \wedge ((\lambda\psi'.\psi'(\lambda x.\lambda y.F(loc(x,y))))(\lambda\gamma.\varphi(\gamma(c))))) \triangleright_{\beta}$$

$$\lambda\varphi.(\exists c)[\text{Computer}(c) \wedge \varphi(\lambda y.F(loc(c,y)))]$$

Interpretation Example

Second application:

$$\lambda\varphi.\llbracket \text{every desk} \rrbracket (\lambda\psi\llbracket \text{computer will be located at it} \rrbracket (\lambda\gamma.\varphi(\psi(\psi)))) \triangleright_{\beta}$$

$$\lambda\varphi.(\lambda\varphi'.\forall x[Desk(x) \rightarrow \varphi'(x)]) (\lambda\psi\llbracket \text{computer located at it} \rrbracket (\lambda\gamma.\varphi(\gamma(\psi))))$$

$$\lambda\varphi.(\forall x)[Desk(x) \rightarrow (\llbracket \text{computer located at it} \rrbracket (\lambda\gamma.\varphi(\gamma(x)))] =$$

$$\lambda\varphi.(\forall x)[Desk(x) \rightarrow ((\lambda\varphi'.(\exists y)(\text{computer}(c) \wedge (\varphi(\lambda x.F(loc(y, x)))))) (\lambda\gamma.\varphi(\gamma(x))))]$$

$$\lambda\varphi.(\forall x)[Desk(x) \rightarrow (\exists y)[\text{Computer}(y) \wedge \varphi(F(loc(x, y)))]]$$

Interpretation Example

Use the trivial continuation $\lambda k.k$:

$$(\lambda\varphi.(\forall x)[Desk(x) \rightarrow (\exists y)[Computer(y) \wedge \varphi(F(loc(x, y)))]])(\lambda k.k) \triangleright_{\beta}$$

$$(\forall x)[Desk(x) \rightarrow (\exists y)[Computer(y) \wedge F(loc(x, y))]]$$

Finally attaching the nominal:

$$@_{now}(\forall x)[Desk(x) \rightarrow (\exists y)[Computer(y) \wedge F(loc(x, y))]]$$

Conclusion

- ▶ We explored how AMR can benefit from formal inference.
- ▶ Described a combination of AMR extensions.
- ▶ Sketched a first-order hybrid logic variant with general proof procedure and model checking methods.
- ▶ Demonstrated how sentences are handled by these methods.
- ▶ Showed how AMR sentences can be translated into *FHTL*

References I

- Girish Shivanand Bhat. 1998. *Tableau-based approaches to model-checking*. Ph.D. thesis, North Carolina State University.
- Patrick Blackburn and Maarten Marx. 2002. Tableaux for quantified hybrid logic. In *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 38–52, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Thomas Bolander and Patrick Blackburn. 2007. Termination for hybrid tableaus. *Journal of Logic and Computation*, 17(3):517–554.
- Thomas Bolander and Patrick Blackburn. 2009. Terminating tableau calculi for hybrid logics extending k. *Electronic Notes in Theoretical Computer Science*, 231:21–39.

References II

- Lucia Donatelli, Michael Regan, William Croft, and Nathan Schneider. 2018. Annotation of tense and aspect semantics for sentential AMR. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 96–108, Santa Fe, New Mexico, USA. Association for Computational Linguistics.
- Melvin Fitting. 1988. First-order modal tableaux. *Journal of Automated Reasoning*, 4(2):191–213.
- Jens Ulrik Hansen. 2007. A tableau system for a first-order hybrid logic. In *Proceedings of the International Workshop on Hybrid Logic (HyLo 2007)*, pages 32–40.
- Hardi Hungar, Orna Grumberg, and Werner Damm. 1995. What if model checking must be truly symbolic. In *Advanced Research Working Conference on Correct Hardware Design and Verification Methods*, pages 1–20. Springer.

References III

James Pustejovsky, Ken Lai, and Nianwen Xue. 2019. Modeling quantification and scope in Abstract Meaning Representations. In *Proceedings of the First International Workshop on Designing Meaning Representations*, pages 28–33, Florence, Italy. Association for Computational Linguistics.