

# Formalizing AMR Inference via Hybrid Logic Tableaux

## CL Masters Thesis Defense

Eli Goldner

July 28, 2021

# Introduction

- ▶ Semantic representation:
  - ▶ Capture meaning of natural language content.
  - ▶ Designed for manipulation via software.
- ▶ Abstract Meaning Representation (AMR):
  - ▶ Graph-based (DAG), nodes are *concepts*, edges are *relations*.
  - ▶ Built on predicative core of a sentence.
  - ▶ Ignores syntactic differences between equivalent sentences.
  - ▶ PropBank framesets are used for concepts (entities, events, properties, states).

# Introduction

(Basic) AMR is reductionistic. This is awesome for:

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

This is not awesome for:

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

# Introduction

- ▶ The trade-off between ease of generation/use and expressivity/inferentiability is as old as computing<sup>1</sup>.
- ▶ AMR has made a choice that works well in data-driven NLP.
- ▶ However AMR can bridge this gap:
- ▶ AMR already does this in a modular way with extensions.
- ▶ Some of these extensions give afford interpretation in first-order logic:
  - ▶ Automated inference for logics is a rich area with lots of tools.
  - ▶ This is where we come in.

---

<sup>1</sup>Possibly as old as cognition.

# Motivation

# Approach

# AMR with Tense and Aspect

# AMR with Scope



# Hybrid Logic

# Hybrid Logic Variants

# First-Order Hybrid Tense Logic

# FHTL Tableau Example

- (1)
- (2)  $@_s(\exists x)[P((\exists y)[f(x, y) = f(y, x)]) \vee \neg(\exists z)[x = z]]$
- (3)  $@_sP((\exists y)[f(s_1, y) = f(y, s_1)]) \vee \neg(\exists z)[s_1 = z]$
- (4)  $@_sP((\exists y)[f(s_1, y) = f(y, s_1)])$        $@_s\neg(\exists z)[s_1 = z]$
- (5)  $@_sPt$        $@_s\neg[s_1 = s_1]$
- (6)  $@_t(\exists y)[f(s_1, y) = f(y, s_1)]$        $@_s[s_1 = s_1]$
- (7)  $\dots$        $\otimes$

# Model Checking Example

- *Every computer will be located at a desk.*

- 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:

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

# 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:

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

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

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

$$D_{yesterday} = \{computer_1, desk_1\}$$

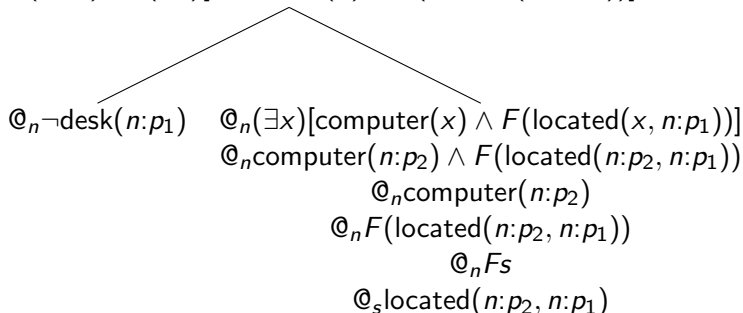
$$D_{now} = \{computer_1, computer_2, desk_1, desk_2, desk_3\}$$

$$D_{tomorrow} = \{computer_1, computer_2, desk_1, desk_2\}$$

# Model Checking Example

$$@_n(\forall y)[\text{desk}(y) \rightarrow (\exists x)[\text{computer}(x) \wedge F(\text{located}(x, y))]]$$

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



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



# Interpretation