Extracting First Order Logic formulas from graphical semantic representations

Rémi de Vergnette, Vincent Tourneur, Maxime Amblard
LORIA, UMR 7503, Université de Lorraine, CNRS, Inria
54000 Nancy, France
remi.de-vergnette@loria.fr, vincent.tourneur@loria.fr, maxime.amblard@univ-lorraine.fr

Abstract

In this paper, we present a method for interpreting YARN structures as logical formulas in a modal first order logic with temporality. YARN is a recent semantic formalism that aims to bridge the gap between graph-based and logic-based semantic representations, providing a flexible and expressive framework for capturing the meaning of natural language utterances. Our approach translates the elements of YARN structures such as predicates, features, into corresponding logical constructs, allowing for an interpretation of the represented meaning. Given that YARN allows underspecified scope, we associate to each YARN structure a set of possible logical interpretations. We account for a range of semantic phenomena, extending beyond ambiguity to capture aspects of dynamic quantification as well. This work contributes to the understanding of the expressive power of graphical semantic representations and their relationship to formal logic.

Keywords: Semantics, Logic, Computational semantics, FOL, modal semantics

Introduction

Semantic formalisms represent a symbolic and explicit approach to capturing and manipulating the meaning of natural language utterances. In contrast to task-oriented representations of semantics, which are generally learned end-to-end using a specific or generic learning task (in the context of transfer learning) and are naturally embedded in continuous vector spaces, semantic formalisms provide explicitly structured representations that aim to capture meaning in a systematic and interpretable way.

Semantic formalisms for Natural Language Processing (NLP) include several advantages. First, their support for compositionality, where the meaning of complex expressions can be derived from the meanings of their parts (Frege, 1891, 1893). Secondly, they allow enhanced explainability, as the structured intermediate representations allows to formalise reasoning in explicit terms, making part of the process more transparent (Nguyen et al., 2025). They also provide easier debiasing, since explicit representations can be made "blind" to certain aspects of the data more systematically (Lobo et al., 2023), and increased frugality, as on one hand semantic representations can be reused across multiple tasks rather than requiring task-specific learning from scratch (Wein and Opitz, 2024), and on the other hand, they tend to build on logical frameworks that enable the use of efficient algorithms for reasoning and inference. Third, semantic annotations have been shown to provide a useful complementary signal to LLMs, for instance in under-resourced languages (Wein, 2025) situations.

The expressivity of structures, relating to what they can represent, provides the absolute bound on the utility of any meaning representation. For instance, when using AMR (Banarescu et al., 2013) alone, which does not fully represent scope, automated natural language inference becomes problematic. Indeed, since two formulas with different scope configurations may have the same representation, it is impossible to process them differently when using AMR alone.

A tension emerges: the more popular formalisms (which are also the ones for which efficient parsers exist) tend to be the least expressive (Pavlova et al., 2023b; Crouch and Kalouli, 2018), while more expressive formalisms, often based on extensions of lambda calculus (Maršík et al., 2021; Amblard and Retoré, 2014) or formal logic (Bos, 1996), are more difficult to parse because of strong structural properties. This implies that obtaining annotations ffor such formalisms generally requires more work.

The introduction of the YARN formalism (Pavlova et al., 2024) is an attempt at resolving this tension by providing a way to represent high-quality structures in terms of semantic, logical, and even syntactic and discursive properties, while maintaining a permissive, scalable, and easy-to-annotate framework for humans. YARN is not alone in extending and refining AMR: UMR (Bonn et al., 2024) represents another such approach, similarly employing a second annotation level to capture higher-level linguistic phenomena. We will be interested in the theoretical properties of YARN structures, more specifically in their expressive power.

Grounding graphical semantic formalisms in formal logic has been explored for related formalisms like AMR (Bos, 2016; Lai et al., 2020), using lambda calculus. (Pavlova et al., 2023a) uses Graph Rewriting Systems for mapping two different semantic

Proceedings of The Seventh International Workshop on Designing Meaning Representations (DMR 2026) @ LREC 2026, pages 102–112
11 May 2026.

formalisms, DRT and AMR. The intermediate scope constraint system we present here is similar to the semantic formalism introduced by (Bos, 1996).

In this paper, we propose how to transform a YARN structure into a set of logical formulas. This provides a way to interpret YARN structures in a logical framework, allowing us to bridge the gap between representations of meaning in YARN and logic formalisms. For the most logically inspired formalisms like DRT (Kamp and Reyle, 1993), there are direct conversions to first order logic. However, given the possibility for underspecification and ambiguity in YARN structures, and the far less constrained structural properties, we will interpret YARN structures as encoding a set of constraints on admissible logical representations.

This is similar to other semantic frameworks such as MRS (Copestake et al., 2005) or Predicate Logic Unplugged (Bos, 1996). Note that since AMR does not encode many logic relevant aspects such as scope, it can be considered an underspecified formalism too. YARN is a superset of AMR and the added features allow more control over the possible interpretations.

After introducing the YARN formalism in Section 1 and the logical framework we use in Section 2, we present in Section 3 our main contribution: we propose an explicit translation from a YARN structure Y to an intermediate representation consisting of a pair (F, R), where F encodes scope dependencies as a labelled forest and R encodes flat predicate-argument relations. We then define a constraint system $C(F, R)$ whose solutions characterize the set of admissible trees compatible with the original YARN structure, thereby making underspecification explicit. For each admissible tree, we provide a compositional interpretation function that yields a modal first-order logic formula with temporality, grounding YARN structures in a logical semantics. We illustrate how the resulting framework captures non-trivial scope interactions and dynamic quantification phenomena on representative examples.

1. YARN

There appears to be a fine line between formalisms that are harder to produce (logic-based) and others that are harder to use for precise tasks (graph-based). Bridging the gap between logic and graphs for semantic annotation is the objective of YARN (Pavlova et al., 2024; Pavlova, 2025).

From graph-based formalisms, especially AMR, YARN adopts the absence of explicit variables in the graphical form of the formalism, a visual approach to annotation, and the use of a graph structure to represent the predicate structure of a sentence. From logical formalisms, YARN adopts the explicit representation of scope through tree-like structures, and the use of features to represent linguistic phenomena and interaction with the predicate part.

YARN adopts a layer-based approach to semantic representation, where there is no single "right" annotation for a sentence, but rather a set of possible annotations, depending on the features modeled. Layers (like tense, aspect, quantification, etc.) can be added or removed depending on the linguistic phenomena of interest. This allows for a flexible representation of meaning that can adapt to different tasks and requirements, as well as to annotator expertise.

YARN does not make assumptions about the underlying modeling of these phenomena and does not commit itself to one particular logical framework: in this sense, it is a "universal" formalism, like DRT.

We first present the formal definition of YARN. As a linguistically inspired framework, YARN encodes linguistic information that goes beyond semantics, providing ways to model information that is irrelevant to the main concern of this paper. We thus focus on a specific subset of elements of YARN structures, which we define below.

1.1. Formal Definition

Following (Pavlova, 2025), a YARN structure is defined as a nine-tuple:

$$\langle S, V, F, D, E, C, L, H, I \rangle$$

where:

- S is a set of vertices representing elementary events
- V is a set of vertices representing predicates and concepts
- F is a set of vertices representing features (tense, aspect, quantification, etc.)
- D is a set of directed edges between pairs of vertices $s_1, s_2 \in S$
- E is a set of directed edges between pairs of elements $v_1, v_2 \in V$
- C is a set of directed edges from a vertex $v \in V$ to a vertex $s \in S$
- L is a set of directed edges from a feature $f \in F$ to a vertex $v \in V$ or an edge $e \in E$
- H is a set of directed edges meeting one of two conditions:
  - start at an F vertex and end in an L or H edge
  - start at an L or H edge and end in a V vertex or an E edge

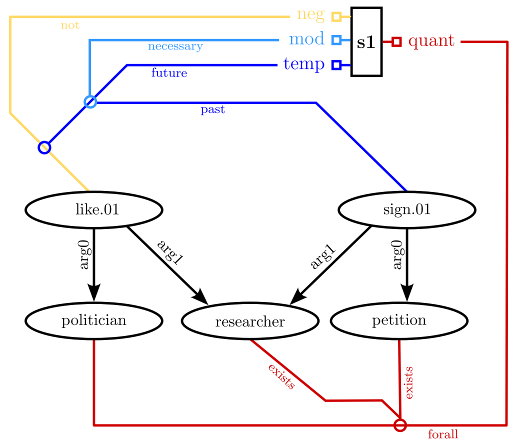

Figure 1: A YARN representation of sentence 1

- I is a set of edges between pairs of elements $v_1, v_2 \in V$

Intuitively, V, E give an AMR-like predicate-argument graph, and F, L, H build a scope-bearing feature structure that can be read as (a set of) operator trees over that graph. D, C specify a higher-level discourse structure. The V and E elements represent the basic predicate structure of a sentence, using PropBank frames for predicates in AMR fashion, and English words for concepts. S elements represent elementary events, linked through discourse relationships modeled by D edges, or linked to V nodes by C edges representing clauses (as in "He said he loved it"). F elements are features allowing the representation of linguistic phenomena like tense, aspect, and quantification. L and H elements allow features to modify predicative elements, with the iterative construction in H enabling scope representation (an H element represents a modification of the element represented by its target, inside the scope of the element represented by its source). I edges provide constraints on the interpretation of V and E elements by specifying set-membership relationships.

1.2. Example

To illustrate YARN's capabilities, let us annotate the sentence:

(1) In the future, politicians won't like researchers who signed a petition.

YARN structures are quite complex when decomposed and described in terms of their atomic elements, but they admit a more readable graphical representation, as shown by Figure 1 (which represents sentence 1).

In this representation, there is one main event taking place, represented by the S node s_1, at the top of the structure, which makes it the root. The predicative structure represents the core meaning of the sentence, consisting of V and E elements. It corresponds to the black graph structure of the figure. A reader trained in AMR will recognize the structure as very similar to the graphical representation of AMR. The predicative structure can be decomposed into V nodes and E edges, where V nodes represent concepts (e.g. "like.01"), and E edges represent relations (e.g. "arg0").

We now restrict the structure to express linguistic phenomena through features and their interactions with the predicative structure. This corresponds to the colored edges and nodes in Figure 1. For features, we show only temporality, negation, modality, and quantification layers. The set of available layers is larger and can be found in (Pavlova, 2025).

The L and H elements specify feature interactions. An L element directly links a feature to the element it modifies, and has a label expressing the kind of modification it performs. For instance, the L edge linking the quantification feature to the "politician" node expresses universal quantification over a variable ranging over politicians. Note that, even though we do not express it explicitly in the graphical representation, we associate a unique identifier to each node, allowing us to introduce a variable for each quantified element.

H elements work in the same way; however, they introduce scope. An H element either:

- links an existing L or H element to an element of the predicative part, such as the edge labelled "exists" in Figure 1.

- links a feature to an existing L or H element. This represents linguistic phenomena taking wide scope over other phenomena. Here, the chain $\xrightarrow{\Box}\xrightarrow{\text{future}}\xrightarrow{\text{not}}$ shows the order in which the modifications should be applied. This ordering represents an event that necessarily will not happen in the future, i.e. an impossible event. A different ordering of the same elements, starting with the "not" feature, represents an event that will not necessarily happen in the future, i.e. an event that might not happen.

As noted earlier, YARN structures can be complex, and not all elements are relevant for the purpose of this paper. We focus on a subset of YARN structures, restricting the features considered to those relevant for logical representation: tense, modality, negation, and quantification. Similarly, we do not treat I elements in this work: as they represent another type of relation between V nodes, we assume we can treat them as E elements. A more precise treatment would require additional assumptions about the semantics we consider, which is out of the scope of this paper. Another feature of YARN that we do not treat in this paper is the possibility to have H edges target E edges, effectively quantifying over relations. We leave this to future work. The annotation layers we consider are sufficiently rich to represent a wide range of linguistic phenomena, but are not the only ones available in YARN. Crucially, we do not treat the aspect layer, which gives information about the internal temporal structure of events, however, the $\prec$ and $\mathcal{O}$ relations we introduce in the logical framework of Section 2 are sufficient to represent such semantic information (Kamp, 2017) , and thus the work presented in this paper can be extended to treat aspect as well. Finally, for simplicity, we only give a complete treatment of annotations with a single root, but as we will see in 3.5, our approach can be easily extended to structures with multiple roots. Even though we do not treat all features and phenomena that can be represented in YARN, the ones we do treat are sufficient to represent a wide range of linguistic phenomena and are the heart of the logical expressivity of YARN and its specification of quantification, which is the main concern of this paper.

2. Modal first order logic with temporality

We now introduce the logical framework we use as a proxy for the semantics of YARN structures.

We take into account modalities, temporality, negation and quantification features. We base ourselves on standard first order logic, adding modal operators $\Box$ and $\Diamond$ for necessity and possibility respectively, following the syntax presented in (Braüner and Ghilardi, 2007). Formal interpretation on modal subordination for semantics have been long studied (Davidson and Harman, 2012; Blackburn and Bos, 2003; Blackburn and Van Benthem, 2007; Qian et al., 2016).

The set of formulas we produce is defined as:

$$\phi ::= A \mid \top \mid \bot \mid p \mid \neg\phi \mid (\phi_1 \wedge \phi_2) \mid (\phi_1 \vee \phi_2) \mid \Diamond(\phi) \mid \Box(\phi) \mid \forall x\ \phi \mid \exists x\ \phi$$

With A the set of atomic formulas defined as:

$$t ::= x \mid c$$
$$A ::= R(t,t) \mid P(t)$$

Where R represents a binary relation symbol, P a predicate symbol, c a constant and x a variable.

In order to treat temporality, we add to the set of binary relations symbols two special temporal relations following the standard approach studied in (Kamp and Reyle, 1993), inspired by (Allen, 1983): $\prec$ for total temporal precedence and $\mathcal{O}$ for temporal overlap, with the following axioms:

- A1: $\forall x \forall y (x \prec y \rightarrow \neg(y\ \mathcal{O}\ x))$
- A2: $\forall x \forall y \forall z (x \prec y \wedge y \prec z \rightarrow x \prec z)$
- A3: $\forall x (x\ \mathcal{O}\ x)$
- A4: $\forall x \forall y (x\ \mathcal{O}\ y \leftrightarrow y\ \mathcal{O}\ x)$
- A5: $\forall x \forall y (x \prec y \rightarrow \neg(x\ \mathcal{O}\ y))$
- A6: $\forall x \forall y \forall z \forall t (x \prec y \wedge y\ \mathcal{O}\ z \wedge z \prec t \rightarrow x \prec t)$
- A7: $\forall x \forall y (x \prec y \vee x\ \mathcal{O}\ y \vee y \prec x)$

The variables in these axioms represents time points or intervals. In order to model the relation between events and the present moment, we also introduce a special constant symbol now, representing the present moment.

This way we can represent temporality in a way compatible with YARN structures, where temporality features are anchored to events.

We now give two examples of formulas representing every day sentences using this framework:

- "Maybe it will rain": $\Diamond(\exists e,\ now \prec e \wedge \text{rain}(e))$
- "John did not see any friend": $\neg\exists e, x, y, b\ e \prec now\ \wedge$ see-01(e) $\wedge$ john(x) $\wedge$ person(y) $\wedge$ befriend(b) $\wedge$ arg0(e,x) $\wedge$ arg1(e,y) $\wedge$ arg0(b,x) $\wedge$ arg1(b,y)

Note that we are using a Neo-Davidsonian (Davidson and Harman, 2012) framework, which comes naturally from the fact that in YARN, we modify each event with individual semantic phenomena.

3. Defining interpretations for YARN structures

3.1. Overview

We now turn to the main goal of this paper: obtaining interpretations as logical formulas for YARN structures. As is the case with our example 1 annotated with the structure presented in Figure 1, many YARN structures are ambiguous, as they do not encode all scopal specifications. We represent meaning as set of possible denotations, in line with (Poesio, 1994).

Our approach works as follows: let Y be a YARN structure. We transform Y into a pair (F, R), where F is a labelled forest (i.e. a set of labelled trees) and R is a set of relations over concept nodes identifiers. This allows us to separate two independent structures: scope dependencies (represented by F) and flat relations between entities (encoded by R). Now R and F give rise to a set of constraints $C(F, R)$ over the set $\mathbb{T}_{\text{all}}$ of trees with the same nodes as F.

We then obtain a set $\mathbb{T}$ of possible tree denotations as trees over nodes of F satisfying $C(F, R)$. Finally, we define an interpretation function, depending on R allowing us to convert every tree of $\mathbb{T}$ to a first-order logical formula with modalities.

For a given YARN structure Y, each element of the intermediate representation $\mathbb{T}$ unambiguously defines a formula. The set of constraints $C(F, R)$ represents both compatibility conditions for preserving scope indications that are already present in Y as well as coherence and linguistically motivated conditions that we shall explicit in 3.3.

3.2. YARN into the forest

Let Y be a YARN structure. We transform Y into a labelled forest F representing every element of the feature structure of Y.

First we introduce a unique identifier for each node of the predicative part (V elements), which will be used as variables in the logical formulas we produce. Here, for each node, we take the first letter of its label, disambiguating "politician and petition" by associating q with "petition" but any unique identifier would do.

This allows to represent the relation part R as follows: we simply extract the set of relations between concept nodes, which are represented by V elements in YARN. We represent them as a set of tuples (s, e_1, e_2) where s is the label of the relation (e.g. "arg0"), and e_1 and e_2 are the unique identifiers of the source and target nodes of the relation respectively. Here $R = \{\text{arg0}(l, p), \text{arg1}(l, r), \text{arg0}(s, r), \text{arg1}(s, q)\}$.

Now to build F we reifiy every H and L elements, representing them as nodes. An H or L element that has a V node as a target is represented as a node with label $Q_{\text{label}}\ e\ p$ where e is a variable representing the target node, $_{\text{label}}$ the label of the H or L element, and p the label of the target node. In that case we say that Q_{label} introduces the variable e. For instance, the L element linking the quantification feature to the "politician" node is represented as a node with label $Q_\forall\ p\ politicians$, which introduces variable p. An H element that has a H or L target is represented as a node with label Q_{label} where $_{\text{label}}$ is the label of the H or L element. For instance, the H element (blue in Figure 1) linking the "temporality" feature to the negation feature L element (yellow in Figure 1 is represented as a node with label Q_{future}.)

Finally, we maintain information about the source and target of H and L elements by representing an H element h_1 that has another H or L element h_2 as a source as a child node of the node representing h_2, and an H or L element h_1 that is the target of another H or L element h_2 as a child node of the node representing h_2. For instance, the

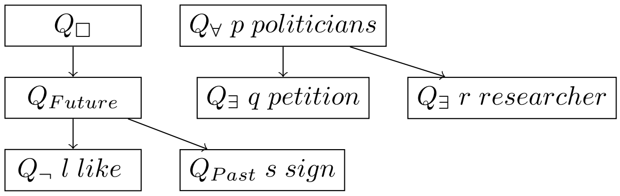

Figure 2: The forest F obtained from the YARN structure of Figure 1

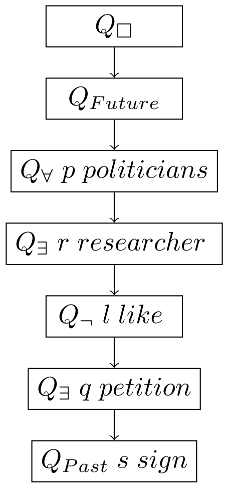

Figure 3: A possible merging for the forest represented in Figure 2

chain $\xrightarrow{\Box} \xrightarrow{\text{future}} \xrightarrow{\text{not}}$ is represented as a tree with root node labelled $Q_\Box$, which has a child node labelled Q_{future}, which itself has a child node labelled $Q_\neg\ l\ like$.

Note that we are conserving the scope information already present in the YARN structure, as the resulting forest is a direct encoding of the relations between H and L elements of the original structure. This achieves the separation of the two structures we mentioned in Section 2, with F representing scope information and R representing relations between entities. Figure 2 shows the forest F obtained from the YARN structure of Figure 1.

3.3. From one forest to many trees

Remember that F encodes general quantification and scope information, and R encodes relations. We now introduce a set of constraints $C(F, R)$ over the set $\mathbb{T}_{\text{all}}$ of trees with the same nodes as F. This allows to describe how we can unify the different trees in F to make a single tree, that we will be able to interpret as a formula. This step is necessary because the original YARN structure does not encode all scopal specifications, and thus we need to introduce constraints to obtain a set of possible trees that are compatible with the original structure.

Let T be a tree in $\mathbb{T}_{\text{all}}$, the set of trees on $\mathbb{V}(F)$, the nodes of F. We say that T satisfies the constraints $C(F, R)$ if and only if the following conditions are satisfied:

- (Compatibility of scopes) For every pair of nodes (e_1, e_2) such that e_2 is a child of e_1 in F, e_2 is a descendant of e_1 in T.
- (Locality of features) In T, every child of a node that doesn't introduce a variable is a child of the same node in F.
- (Participant before event) For every relation $r(e_1, e_2)$ in R, if e_1 is introduced by a node n_1 and e_2 is introduced by a node n_2, n_2 is an ancestor of n_1 and every node that is on the path between n_2 and n_1 and that introduces a variable e_3 is such that there exists a relation $r'(e_3, e_2)$ or $r'(e_1, e_3)$ in R.

(Compatibility of scopes) condition plainly describes that we may not break the scope relations already present in F when building a tree T.

(Locality of features) condition describes that feature acting over other features do not modify wider scope features.

(Participant before event) condition is a linguistically motivated constraint to restrict the number of possible trees, and it may be relaxed. It says that we introduce events after their participants, and that we do not introduce any variable in the scope of the event's participants before introducing the event itself, unless it is another event relating the same participants.

$\mathbb{T}$ can be constructed by enumerating all possible trees that satisfy the (Compatibility of scopes) and (Locality of features) conditions. During the enumeration, at each step where we add a node, we check if its ancestors satisfy the constraints (Participant before event), and if it is not the case, we discard all the solutions that can be obtained from the current state. Figure 3 shows a possible tree obtained by merging the two trees of the forest F while respecting the constraints we just introduced.

3.4. Interpretation of trees over formulas

Having obtained the set $\mathbb{T}$ of trees over $\mathbb{V}(F)$ satisfying $C(F, R)$, we now define an interpretation function allowing us to convert every element T of $\mathbb{T}$ to a first order logical formula with modality.

Since T only represents the structure of a possible formula, we first need to enrich it with elements of R in order to inject the full semantic content of the structure. For each node of T, we compute the list of relations in R that are between variables introduced by this node or its ancestors. If this list is not empty, we append a child node that contains

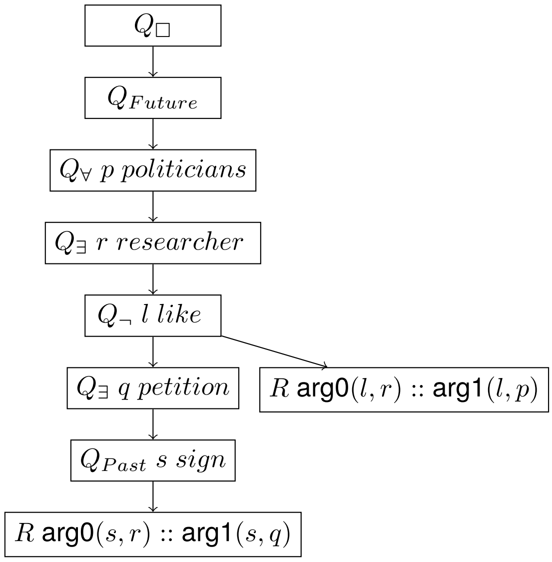

Figure 4: The merging of Figure 3 enriched with the relations in R

this list. Figure 4 shows the tree obtained after adding these relation nodes. We write the result of this operation Join(T, R).

Then we can define an interpretation function $[\![\cdot]\!]_\tau$ that allows us to convert every element of $\mathbb{T}$ to a first order logical formula with modality. To achieve this, we need a linear representation of the trees, which will be the input of the function $[\![\cdot]\!]_\tau$. This representation is defined in Figure 5. **S** represents a tree, and each possible rule encodes the type of the root node. For example, the root node of the tree in Figure 4 is encoded by rule 5 ($Q_\Box$ **F**). Each possible node type of **S** includes an **S**, which is a list of trees, to represent all child trees (except nodes created with rule 2, since they are always leaf nodes). **A** is a list of relations, which is used by nodes of type R (rule 2). The symbol :: serves as a separator between elements in both lists. The full translation of the tree in Figure 4 is Join$(T, R) =$ $(Q_\Box\ (Q_{Future}\ (Q_\forall\ p\ politicians\ (Q_\exists\ r\ researcher$ $(Q_\neg\ l\ like\ (R$ arg0(l, r) :: arg1(l, p) :: [])
$(Q_\exists\ q\ petition$
$(Q_{Past}\ s\ sign\ (R$ arg0(s, r) :: arg1(s, q) :: [])
:: []) :: []) :: []) :: []) :: []) :: []) :: [])

The function $[\![\cdot]\!]_\tau$ is then defined for each rule of the tree grammar. The full definition is given in Figure 6. τ represents the temporal context in which the formula is evaluated, and it is used to interpret temporal operators. This provides a unique interpretation of every tree in $\mathbb{T}$, and thus for the original YARN structure Y given the solution to the constraints $C(F, R)$ provided by $\mathbb{T}$, by computing $[\![$ Join(T,R) $]\!]_{now}$.

The result for the tree of Figure 4

$$
\begin{array}{rlr}
\textbf{S} ::= & (R\ \textbf{A}) & (2)\\
\mid & (Q_\forall\ e\ p\ \textbf{F}) & (3)\\
\mid & (Q_\exists\ e\ p\ \textbf{F}) & (4)\\
\mid & (Q_\Box\ \textbf{F}) & (5)\\
\mid & (Q_\Box\ e\ p\ \textbf{F}) & (6)\\
\mid & (Q_\Diamond\ \textbf{F}) & (7)\\
\mid & (Q_\Diamond\ e\ p\ \textbf{F}) & (8)\\
\mid & (Q_\neg\ \textbf{F}) & (9)\\
\mid & (Q_\neg\ e\ p\ \textbf{F}) & (10)\\
\mid & (Q_{Past}\ \textbf{F}) & (11)\\
\mid & (Q_{Past}\ e\ p\ \textbf{F}) & (12)\\
\mid & (Q_{Present}\ \textbf{F}) & (13)\\
\mid & (Q_{Present}\ e\ p\ \textbf{F}) & (14)\\
\mid & (Q_{Future}\ \textbf{F}) & (15)\\
\mid & (Q_{Future}\ e\ p\ \textbf{F}) & (16)\\
\textbf{F} ::= & [\ ]_F & (17)\\
\mid & \textbf{S} :: \textbf{F} & (18)\\
\textbf{A} ::= & [\ ]_A & (19)\\
\mid & (s, e, f) :: \textbf{A} & (20)
\end{array}
$$

In all rules where they appear, e and f are variable names, p is a predicate, and s is a relation.

Figure 5: The grammar which defines the linear representation of trees in $\mathbb{T}$

$$
\begin{array}{rr}
[\![\ [\ ]_A\ ]\!]_\tau \rightarrow \top & (21)\\
[\![\ (s, e_1, e_2) :: al\ ]\!]_\tau \rightarrow s(e_1, e_2) \wedge [\![\ al\ ]\!]_\tau & (22)\\
[\![\ [\ ]_F\ ]\!]_\tau \rightarrow \top & (23)\\
[\![\ t :: f\ ]\!]_\tau \rightarrow [\![\ t\ ]\!]_\tau \wedge [\![\ f\ ]\!]_\tau & (24)\\
[\![\ R\ al\ ]\!]_\tau \rightarrow [\![\ al\ ]\!] & (25)\\
[\![\ Q_\forall\ e\ p\ f\ ]\!]_\tau \rightarrow \forall e, p(e) \Rightarrow [\![\ f\ ]\!]_\tau & (26)\\
[\![\ Q_\exists\ e\ p\ f\ ]\!]_\tau \rightarrow \exists e, p(e) \wedge [\![\ f\ ]\!]_\tau & (27)\\
[\![\ Q_\Box f\ ]\!]_\tau \rightarrow \Box([\![\ f\ ]\!]_\tau) & (28)\\
[\![\ Q_\Box\ e\ p\ f\ ]\!]_\tau \rightarrow \Box(\exists e, p(e) \wedge [\![\ f\ ]\!]_\tau) & (29)\\
[\![\ Q_\Diamond\ f\ ]\!]_\tau \rightarrow \Diamond([\![\ f\ ]\!]_\tau) & (30)\\
[\![\ Q_\Diamond\ e\ p\ f\ ]\!]_\tau \rightarrow \Diamond(\exists v, p(v) \wedge [\![\ f\ ]\!]_\tau) & (31)\\
[\![\ Q_\neg f\ ]\!]_\tau \rightarrow \neg([\![\ f\ ]\!]_\tau) & (32)\\
[\![\ Q_\neg\ e\ p\ f\ ]\!]_\tau \rightarrow \neg(\exists e, p(e) \wedge [\![\ f\ ]\!]_\tau) & (33)\\
[\![\ Q_{Past}\ f\ ]\!]_\tau \rightarrow \exists e, e \prec \tau\ \wedge [\![\ f\ ]\!]_e & (34)\\
[\![\ Q_{Past}\ e\ p\ f\ ]\!]_\tau \rightarrow \exists e, p(e) \wedge e \prec \tau\ \wedge [\![\ f\ ]\!]_e & (35)\\
[\![\ Q_{Present}\ f\ ]\!]_\tau \rightarrow \exists e, \tau\ \mathcal{O}\ e\ \wedge [\![\ f\ ]\!]_e & (36)\\
[\![\ Q_{Present}\ e\ p\ f\ ]\!]_\tau \rightarrow \exists e, p(e) \wedge \tau\ \mathcal{O}\ e\ \wedge [\![\ f\ ]\!]_e & (37)\\
[\![\ Q_{Future}\ f\ ]\!]_\tau \rightarrow \exists e, \tau \prec e \wedge [\![\ f\ ]\!]_e & (38)\\
[\![\ Q_{Future}\ e\ p\ f\ ]\!]_\tau \rightarrow \exists e, p(e) \wedge \tau \prec e \wedge [\![\ f\ ]\!]_e & (39)
\end{array}
$$

In rules 34, 36 and 38, e is a fresh variable name.

Figure 6: The definition of function $[\![\cdot]\!]_\tau$ for every rule of the tree grammar

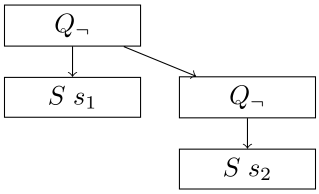

Figure 7: A consequence relation between two S nodes modelled using $Q_\neg$ nodes

is $[\![$ Join(T,R) $]\!]_{now} = \Box(\exists e, now \prec e \wedge (\forall p, politician(p) \Rightarrow (\exists r, researcher(r) \wedge (\neg(\exists l, like(l) \wedge (\exists q, petition(q) \wedge (\exists s, sign(s) \wedge e \prec s \wedge \text{arg0}(s, r) \wedge \text{arg1}(s, q)) \wedge \text{arg0}(l, r) \wedge \text{arg1}(l, p)))))))$

The other possible tree gives rise to the reading where the "petition" q variable is introduced before the "researcher" r variable, which corresponds to the reading where there is a specific petition whose signing makes politicians dislike researchers, which is a weaker but valid reading of the original sentence. We now follow the approach of (Bos, 1996) and define a semantic for YARN by defining the interpretation of Y as the set of all pairs consisting of a tree T in $\mathbb{T}$ and the truth value of $[\![$ Join(T, R) $]\!]_{now}$.

In plain words, we interpret a YARN structure as a set of possible interpretations together with their truth values.

3.5. Extension to complex structures

In this section, we show how to extend the approach we just described to YARN structures involving several S nodes, linked together by elements of type D. This is important, as many YARN structures involve several S nodes, and we want to be able to obtain interpretations for such structures as well. Remember that S nodes represent main events, and that D represent discourse relations. Different YARN substructures may share the same V nodes, thus we can't treat them as independent structures, and we need to take into account the relations between them when building the forest F and the set of relations R. We will show how to treat the "consequence" relation, which we model as implication.

We extend the forest building step of 3.2: we first build a forest for each S node independently, and then add a new node labeled $Svar$ for each S node, which introduces a variable representing the event corresponding to this S node. Writing $a \Rightarrow b$ as $\neg(a \wedge \neg b)$, we add two $Q_\neg$ nodes, in the configuration shown in Figure 7. Note that if we wished to consider that the consequence relation induces a temporal precedence phenomenon, we could have done so by the mean of an additional

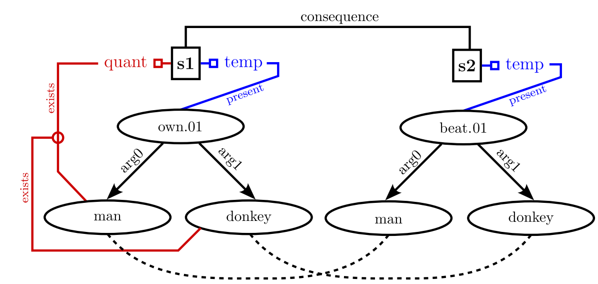

Figure 8: A possible YARN structure for the sentence “Every farmer who owns a donkey beats it”

Q_{Future} node.

We can apply the same steps as before to obtain a set of trees $\mathbb{T}$, and then apply the interpretation function to obtain a set of formulas. We only need to update the grammar 5 to allow for the new elements we introduce, and provide a suitable interpretation for them: **S** $::=$ $(S\ e$ **F**$)$ and $[\![\ S\ e\ f\ ]\!]_\tau \rightarrow [\![\ f\ ]\!]_e$. As an example of the expressivity of this system, we show how this allows us to obtain a correct interpretation for donkey style structures. Such constructions are well known to pose profound challenges related to compositionality, quantification and coreference (Kamp, 1981; Partee et al., 1984; Kanazawa, 1994). Since YARN does not encode a coreference resolution mechanism, the primary aim of this example is to demonstrate that dynamic quantification phenomena are naturally handled by our approach, yielding correct interpretations for such sentences without requiring any additional machinery.

Consider the following sentence:

(40) If a man owns a donkey, he beats it.

We propose a YARN annotation for this sentence, which is shown in Figure 8. For readability purposes, we have duplicated the nodes corresponding to “man” and “donkey”, but there is only one occurrence of each node in the actual structure. This is represented by the dashed lines in the bottom of Figure 8.

Applying the approach presented in 3.2, we obtain a forest F, represented graphically in Figure 9. As for relations, we have $R = \{\text{arg0}(o,m), \text{arg1}(o,d), \text{arg0}(b,m), \text{arg1}(b,d)\}$, where we introduced the variables m and d to represent the “man” and “donkey” nodes respectively, and o and b for the “own” and “beat” nodes respectively.

Then, after applying the approach of 3.3, we obtain a set of trees $\mathbb{T}$, which is reduced to a single tree T in this particular case. This tree is represented graphically in Figure 10.

Finally, using the rewriting system of Subsection 3.4, we obtain the following formula: $[\![\ \text{Join}(T,R)\ ]\!]_{now} = \neg\exists x,y,o\Big(man(x) \wedge donkey(y) \wedge own(o) \wedge o\ \mathcal{O}\ now \wedge \text{arg0}(o,x) \wedge \text{arg1}(o,y) \wedge \neg\exists b\big(beat(b) \wedge b\ \mathcal{O}\ o \wedge \text{arg0}(b,x) \wedge \text{arg1}(b,y)\big)\Big)$ which says “There is no man and no donkey such that that man owns that donkey and does not beat it”, which is a correct interpretation for the sentence 40.

Using the equivalences $\neg\exists x, P(x) \equiv \forall x, \neg P(x)$ and $\neg(A \wedge \neg B) \equiv A \Rightarrow B$, we can rewrite this formula in a more natural form as follows: $\forall x,y,o\Big(man(x) \wedge donkey(y) \wedge own(o) \wedge o\ \mathcal{O}\ now \wedge \text{arg0}(o,x) \wedge \text{arg1}(o,y) \Rightarrow \exists b\big(beat(b) \wedge b\ \mathcal{O}\ o \wedge \text{arg0}(b,x) \wedge \text{arg1}(b,y)\big)\Big)$

Conclusion

YARN is a linguistic formalism grounded in logic, both in its underlying motivations and in its formal design. We have shown that YARN structures admit a systematic translation into logical formulas. This suggests that YARN can be regarded as a semantic representation in the strict sense, in so far as truth conditions can be extracted from its structures.

While the presence of structural ambiguity may appear problematic at first, it can also be viewed as an expressive feature, allowing for the representation of complex quantificational patterns, by describing a set of possible interpretations. This is a natural way to account for the ambiguity of natural language, and it allows to represent complex scopal interaction, however, it is not the only solution, and one may argue that the absence of scope specification between two phenomena is not always a sign of ambiguity, but rather a sign of independence between the two phenomena. In that case, we could consider that a structure denotes a single formula, and that the absence of scope specification between two phenomena is a sign of independence between them, which can be modeled by other logical means (Hintikka, 1979), such as the use of Henkin quantifiers (Henkin and Karp, 1965), or Independance Friendly Logic (Hintikka and Sandu, 1989). This would allow to explore the use of YARN as an interlingua for semantic representation, which offer interesting perspectives for symbolic NLP applications, notably due to its visual characteristics.

The scope constraint system and the conversion algorithm presented here are closely related to previous work, in particular that of (Bos, 1996), but our approach differs in its algorithmic nature, the explicit separation of scope and relation structures, and

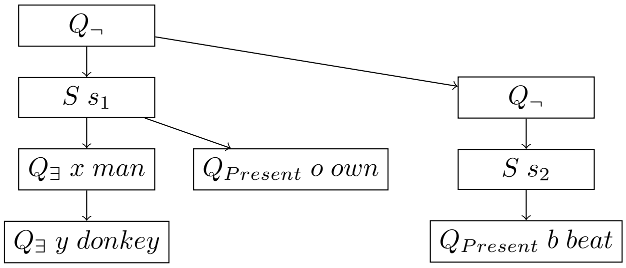

Figure 9: The forest obtained from the YARN structure of Figure 8

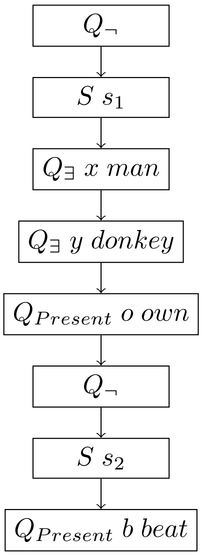

Figure 10: The only possible tree obtained by merging the forest in Figure 9

our commitment to a specific logical framework.

We also introduce a linguistically motivated constraint to restrict the number of possible trees. Note that we are only assuming logical interpretations for the action of features at the very last step: as such, providing alternative interpretations doesn't require any change in the previous steps. For instance, evolving our interpretation for trees by adding a context, as is done in (De Groote, 2006), would only require a change in the definition of $[\![\cdot]\!]_\tau$. This would allow to give a compositional interpretation of YARN structure linked by discourse relation, without needing to merge them into a single structure, assuming they do not share nodes.

Ethical considerations

This study raises no specific ethical concerns beyond those inherent to semantic representation itself. The primary limitations relate to ontological bias in meaning representations, largely attributable to the disproportionate prevalence of English in available linguistic resources and work.

Acknowledgements

We would like to thank Amandine Decker for her attentive and considerate help in proofreading the document. We also thank the reviewers for their suggestions and for helping us broaden our knowledge of previous work on these subjects.

4. Bibliographical References

James F. Allen. 1983. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26:832–843.

Maxime Amblard and Christian Retoré. 2014. Partially Commutative Linear Logic and Lambek Calculus with Product: Natural Deduction, Normalisation, Subformula Property. *IfColog Journal of Logics and their Applications (FLAP)*, 1(1):53–94.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for Sembanking.

Patrick Blackburn and Johan Bos. 2003. Computational semantics. *Theoria: An International Journal for Theory, History and Foundations of Science*, pages 27–45.

Patrick Blackburn and Johan Van Benthem. 2007. 1 modal logic: a semantic perspective. In *Studies in logic and practical reasoning*, volume 3, pages 1–84. Elsevier.

Julia Bonn, Matthew J. Buchholz, Jayeol Chun, Andrew Cowell, William Croft, Lukas Denk, Sijia Ge, Jan Hajič, Kenneth Lai, James H. Martin, Skatje Myers, Alexis Palmer, Martha Palmer, Claire Benet Post, James Pustejovsky, Kristine Stenzel, Haibo Sun, Zdeňka Urešová, Rosa Vallejos, Jens E. L. Van Gysel, Meagan Vigus, Nianwen Xue, and Jin Zhao. 2024. Building a broad infrastructure for uniform meaning representations. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2537–2547, Torino, Italia. ELRA and ICCL.

Johan Bos. 1996. Predicate logic unplugged.

Johan Bos. 2016. Squib: Expressive power of Abstract Meaning Representations. *Computational Linguistics*, 42(3):527–535.

Torben Braüner and Silvio Ghilardi. 2007. 9 first-order modal logic. In Patrick Blackburn, Johan Van Benthem, and Frank Wolter, editors, *Handbook of Modal Logic*, volume 3 of *Studies in Logic and Practical Reasoning*, pages 549–620. Elsevier.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A Sag. 2005. Minimal recursion semantics: An introduction. *Research on language and computation*, 3(2):281–332.

Richard Crouch and Aikaterini-Lida Kalouli. 2018. Named graphs for semantic representation. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 113–118, New Orleans, Louisiana. Association for Computational Linguistics.

Donald Davidson and Gilbert Harman. 2012. *Semantics of natural language*, volume 40. Springer Science & Business Media.

Philippe De Groote. 2006. Towards a montagovian account of dynamics. In *Semantics and linguistic theory*, pages 1–16.

Gottlob Frege. 1891. Function and concept. In Peter Geach and Max Black, editors, *Translations from the Philosophical Writings of Gottlob Frege*, pages 21–41. Basil Blackwell, Oxford. Original work published 1891.

Gottlob Frege. 1893. *Grundgesetze der Arithmetik*, volume 1–2. Hermann Pohle, Jena.

Leon Henkin and Carol R. Karp. 1965. Some remarks on infinitely long formulas. *Journal of Symbolic Logic*, 30(1):96–97.

Jaakko Hintikka. 1979. Quantifiers vs. quantification theory. In *Game-Theoretical Semantics: Essays on Semantics by Hintikka, Carlson, Peacocke, Rantala, and Saarinen*, pages 49–79. Springer.

Jaakko Hintikka and Gabriel Sandu. 1989. Informational independence as a semantical phenomenon. In Jens Erik Fenstad, Ivan T. Frolov, and Risto Hilpinen, editors, *Logic, Methodology and Philosophy of Science VIII*, volume 126 of *Studies in Logic and the Foundations of Mathematics*, pages 571–589. Elsevier.

Hans Kamp. 1981. A theory of truth and semantic representation. In Jeroen Groenendijk, Theo Janssen, and Martin Stokhof, editors, *Formal Methods in the Study of Language*, volume 135 of *Mathematical Centre Tracts*, pages 277–322. Mathematical Centre, Amsterdam.

Hans Kamp. 2017. Events, discourse representations and temporal reference. *Semantics and Pragmatics*, 10:2–1.

Hans Kamp and Uwe Reyle. 1993. *From discourse to logic: Introduction to modeltheoretic semantics of natural language, formal logic and discourse representation theory*, volume 42. Springer Science & Business Media.

Makoto Kanazawa. 1994. Weak vs. strong readings of donkey sentences and monotonicity inference in a dynamic setting. *Linguistics and philosophy*, 17(2):109–158.

Kenneth Lai, Lucia Donatelli, and James Pustejovsky. 2020. A continuation semantics for Abstract Meaning Representation. In *Proceedings of the Second International Workshop on Designing Meaning Representations*, pages 1–12, Barcelona Spain (online). Association for Computational Linguistics.

Paula Reyero Lobo, Enrico Daga, Harith Alani, and Miriam Fernandez. 2023. Semantic web technologies and bias in artificial intelligence: A systematic literature review. *Semantic Web*, 14(4):745–770.

Jirka Maršík, Maxime Amblard, and Philippe de Groote. 2021. Introducing (| λ |), a λ-calculus for effectful computation. *Theoretical Computer Science*, 869:108–155.

Giang Nguyen, Ivan Brugere, Shubham Sharma, Sanjay Kariyappa, Anh Totti Nguyen, and Freddy Lecue. 2025. Interpretable LLM-based table question answering. *Transactions on Machine Learning Research*.

Barbara H. Partee, Alice ter Meulen, and Robert E. Wall. 1984. *Mathematical Methods in Linguistics*. D. Reidel Publishing Company, Dordrecht.

Siyana Pavlova. 2025. *Tools and methods for semantically annotated corpora*. Ph.D. thesis, Université de Lorraine.

Siyana Pavlova, Maxime Amblard, and Bruno Guillaume. 2023a. Bridging Semantic Frameworks: mapping DRS onto AMR. In *Proceedings of The*

15th International Conference on Computational Semantics (IWCS 2023), Nancy, France.

Siyana Pavlova, Maxime Amblard, and Bruno Guillaume. 2023b. Structural and global features for comparing semantic representation formalisms. In *Proceedings of the Fourth International Workshop on Designing Meaning Representations*, pages 1–12, Nancy, France. Association for Computational Linguistics.

Siyana Pavlova, Maxime Amblard, and Bruno Guillaume. 2024. YARN is All You Knit: Encoding Multiple Semantic Phenomena with Layers. In *Proceedings of the Fifth International Workshop on Designing Meaning Representations @ LREC-COLING 2024*, pages 66–76, Torino, Italia. ELRA and ICCL.

Massimo Poesio. 1994. *Discourse interpretation and the scope of operators*. Ph.D. thesis. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Dernière mise à jour - 2023-07-26.

Sai Qian, Philippe de Groote, and Maxime Amblard. 2016. Modal Subordination in Type Theoretic Dynamic Logic. *Linguistic Issues in Language Technology*, 14((1)):1–39.

Shira Wein. 2025. Can uniform meaning representation help GPT-4 translate from indigenous languages? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 278–285, Vienna, Austria. Association for Computational Linguistics.

Shira Wein and Juri Opitz. 2024. A survey of amr applications. In *EMNLP*, pages 6856–6875.