Predicates and entities in Abstract Meaning Representation

Antoine Venant
OLST, Université de Montréal
antoine.venant@umontreal.ca

François Lareau
OLST, Université de Montréal
francois.lareau@umontreal.ca

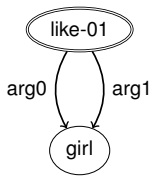
Abstract

Nodes in Abstract Meaning Representation (AMR) are generally thought of as neo-Davidsonian entities. We review existing translation into neo-Davidsonian representations and show that these translations inconsistently handle copula sentences. We link the problem to an asymmetry arising from a problematic handling of words with no associated PropBank frames for the underlying predicate. We introduce a method to automatically and uniformly decompose AMR nodes into an entity-part and a predicative part, which offers a consistent treatment of copula sentences and quasi-predicates such as *brother* or *client*.

1 Introduction

Over the past decade, graph-based semantic representation formalisms have gained a lot of attention, with Abstract Meaning Representation (AMR) arguably leading the trend (Banarescu et al., 2013; Knight et al., 2021). AMR takes a dependency approach to meaning representation, using labeled directed acyclic graphs (incidentally called Abstract Meaning Representations). Labeled graph nodes represent *concepts*, while labeled directed edges represent the *roles* that concepts play in relation to others. For instance, in the sentence *A girl likes herself*, the entity concepts denoted by the noun *girl* plays two roles (agent and theme, Parsons, 1990) in the eventuality concept denoted by the verb *likes*, which is represented by the AMR in figure 1, both as (a) a graph and (b) a tree in PENMAN notation (Mann, 1983). The root of the graph is indicated by double boundaries. Note the use of co-indexed variables in the tree to express graph reentrance. AMR relies whenever possible on PropBank's frames (Palmer et al., 2005) for thematic role labeling (in this case, the frame *likes-01*), hence the use of arg0 and arg1 rather than *agent* and *theme*.

The root of an AMR serves as "a rudimentary representation of overall focus" (Banarescu et al., 2019). Following AMR guidelines, an existential sentence like *There is a girl who likes herself* or a phrase like *a girl who likes herself* would be associated the same representation, shown in figure 2, which differs from the one in figure 1 only by its root. Figure 2 also illustrates how inverse roles like arg0-of are used to unfold a directed graph as a tree from a designated root.

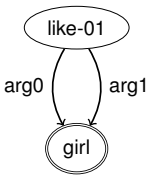


(a) Graph

```
(l / like-01
 :arg0 (x / girl)
 :arg1 x
)
```

(b) Tree

Figure 1: AMR for *A girl likes herself*



(a) Graph

```
(x / girl
 :arg0-of
  (l / like-01
   :arg1 x
  )
)
```

(b) Tree

Figure 2: AMR for *There is a girl who likes herself*

AMR, like other formalisms historically linked to natural language generation (*e.g.*, Meaning-Text Theory, Mel'čuk, 1973, 2016), departs from logical meaning representations insofar as it comes with no notion of bound variable or scope. The semantic treatment of determiners and adverbs closely mirrors their treatment in dependency syntax, as they simply introduce a polarity or quantity role in the concept denoted by the word they modify. Overall, AMR aims at a transparent representation of predicate-argument structure independent from syntactic contingencies[1] and leaves aside other as-

[1] For instance, referring to an event with a verb or a noun (possibly combined with a light verb).

pects of logical structure, most notably scoping phenomena induced by quantification, negation or modality. As a result, AMR offers a simplified formal apparatus for meaning representation, yielding better consistency among annotators, and effective comparison metrics for comparing semantic annotations.[2] These operational and computational benefits however come at the cost of lacking a model-theory (or a proof-theory, for that matter). This might be unsatisfying empirically or theoretically: empirically, because a formalisation of entailment or equivalence between different AMRs might help downstream tasks or resolve annotation conflicts, and theoretically, because one might want semantic descriptions to describe the recursive relationship between semantic components and the world to count as more than paraphrases of the original sentences (Davidson, 1967). For these reasons, translations of large fragments of AMR into logic have been proposed (Bos, 2016; Lai et al., 2020).

While discussions of AMR's relationship to logical or model-theoretic approaches mostly revolve around questions of quantifier scope, in this paper, we are instead concerned with the **type** of object that AMR nodes denote. Our aim is to answer the following: do AMR nodes always denote entities (objects, events, or states), or do they sometimes denote properties or propositions? Do they denote several of these things at once? Can one **consistently** assign a denotation of a fixed type to a node?

We will argue that the answer to these questions is less clear-cut than AMR's general framing as a "simplified, standard neo-Davidsonian semantics" (Banarescu et al., 2019) or existing translations into logic might suggest. There is a potential difficulty which might prevent us from systematically thinking of AMR nodes as objects, events or states. AMR merges two distinct terms of classical logic into a single node (or rather, a single attachment point): predicate and variable symbols. The logical counterpart to a node labeled 'cat' involves intuitively a formula like $\mathsf{cat}(x)$, combining a predicate symbol cat and a variable x. By *merging*, we do not mean that AMR does not *display* these two components separately (in fact, it does, since the entity arguably corresponds to the node itself, and the predicate to its label).[3] What we mean, is that AMR merges the possibilities to further *refer* to the predicate, or the variable. In other words, a node features only one attachment point for two distinct components of meaning.[4] The situation is different in, say, Church's theory of simple types (commonly used in semantics, in conjunction with the neo-Davidsonian approach), where a term like $cat(x)$ results from the combination of two different terms in the lexicon, which nothing refrains from occurring separately as arguments of other terms. One is of type $\langle e,t\rangle$ ($\lambda x\,\mathsf{cat}(x)$, generally modeled as originating from the noun), and one of type e (x, generally modeled as originating from the determiner, jointly with a binding quantifier). We will assess whether this difference challenges a denotational semantics for AMR. We will base our investigation on two types of evidence: the logical translation proposed by Lai et al. (2020) and intuitions based on annotated sentences in AMR corpora and guidelines (Knight et al., 2021; Banarescu et al., 2019).

2 Graph nodes as entities

As mentioned above, AMR claims to implement a simplified neo-Davidsonian semantics, which naturally suggests interpreting nodes as individuals or eventualities: "We do not point to an element in an AMR and say 'that is a noun' or 'that is a verb'. Rather, we say 'that is an object' or 'that is an event"' (Banarescu et al., 2019). Bos (2016) and Lai et al. (2020) formalize the connection with systematic translations from AMR into symbolic logic. We will take the latter (which in many ways constitutes a refinement of the former) as a starting point to examine the denotation of AMR elements.

By definition, a *compositional* semantics for AMR needs syntactic composition rules to operate. Since the treatment of universal quantification or negation has no incidence on our discussion, we restrict ourselves to what Bos (2016) calls "basic" AMRs, the syntax of which is described by the BNF grammar below.

$$\mathbf{A} ::= c \mid x \mid (\mathbf{N})$$
$$\mathbf{N} ::= x/P \mid \mathbf{N}{:}r\,\mathbf{A} \mid \mathbf{N}{:}r^{-1}\,\mathbf{A}$$

Non-terminals are in bold sans. x, c, P and r are meta-variables. x ranges over variable

[2] Importantly, metrics such as SMATCH approximating graph homomorphism through comparison of ⟨start node, label, end node⟩ triples.

[3] AMR authors also introduce an alternative logical notation $\exists x\mathsf{instance}(x,\mathsf{cat})$ employing an *instance* relation, which they claim to be equivalent to the graph.

[4] This seems true even under the *instance*-based representations because no other role than *instance* ever attach to the target node of an *instance* edge, to our knowledge.

symbols $\{x, y, \dots\}$, c over constant sequences {"M. Krupps", "Obama", ...}, P over node labels $\{\text{like-01}, \text{girl}, \dots\}$, and r over (non-inverse) roles $\{\text{arg0}, \text{arg1}, \text{domain}, \dots\}$. r^{-1} describes the inverse of role r (arg0^{-1} = arg0-of, domain^{-1} = mod, ...). Finally, ε denotes the empty sequence. Both Bos (2016) and Lai et al. (2020) syntactically distinguish "projective" and "assertive" nodes to provide a sound treatment of reentrance. We simplify this by systematically interpreting arguments to an inverse role as assertive and arguments to a standard role as projective without additional syntax.[5]

Lai et al. (2020) provide a continuation-style compositional semantics for AMR. The semantic composition rules are given below.

$$[\![c]\!] = \lambda f\, f(c)$$
$$[\![x]\!] = \lambda f\, f(x)$$
$$[\![(\mathbf{N})]\!] = [\![\mathbf{N}]\!]$$
$$[\![x/P]\!] = \lambda f\, \exists x P(x) \wedge f(x)$$
$$[\![\mathbf{N}\!:\!r\,\mathbf{A}]\!] = \lambda f\, [\![\mathbf{A}]\!](\lambda m\, [\![\mathbf{N}]\!](\lambda n\, r(n,m) \wedge f(n)))$$
$$[\![\mathbf{N}\!:\!r^{-1}\,\mathbf{A}]\!] = \lambda f\, [\![\mathbf{N}]\!](\lambda n\, [\![\mathbf{A}]\!](\lambda m\, r(m,n) \wedge f(n)))$$

Figure 4 illustrates the syntax of the AMR of figure 2 (*There is a girl who likes herself*). Figure 3 shows how the semantics of this example unpacks.

$$[\![x]\!] = \lambda f\, f(x)$$
$$[\![a/\text{like-01}]\!] = \lambda f\, \exists a\, \text{like-01}(a) \wedge f(a)$$
$$[\![x/\text{girl}]\!] = \lambda f\, \exists x\, \text{girl}(x) \wedge f(x)$$
$$[\![a/\text{like-01}]_{\mathbf{N}}\ \text{:arg1}\ [x]_{\mathbf{A}}]\!]$$
$$= \lambda f\, [\![x]\!](\lambda m\, [\![\text{a/like-01}]\!](\lambda n\, \text{arg1}(n,m) \wedge f(n)))$$
$$= \lambda f\, \exists a\, \text{like-01}(a) \wedge \text{arg1}(a,x) \wedge f(a)$$
$$[\![[x/\text{girl}]_{\mathbf{N}}\ \text{:arg0-of}\ [(\text{a/like-01 :arg1}\ x)]_{\mathbf{A}}]\!]$$
$$= \lambda f\, [\![x/\text{girl}]\!](\ \lambda n\, [\![(\text{a/like-01 :arg1}\ x)]\!](\lambda m\, \text{arg0}(m,n) \wedge f(n)))$$
$$= \lambda f\, \exists x\, \text{girl}(x) \wedge \exists a\, \text{like-01}(a) \wedge \text{arg1}(a,x) \wedge \text{arg0}(a,x) \wedge f(x)$$

Figure 3: Logical interpretation of (x/girl :arg0-of (a/like-01 :arg1 x))

The above rules interpret AMR as trees rather

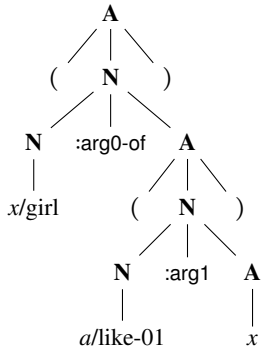


Figure 4: Syntax of (x/girl :arg0-of (a/like-01 :arg1 x))

than graphs. Two AMRs differing only by their root will generally receive distinct denotations. For instance the AMRs from figures 1 and 2 contribute equivalent propositions, but they would pass on different entities as argument to incoming roles (respectively, the girl and the liking). This also means that graph nodes have a dual denotation, because they generally assume two different forms in the corresponding tree. For each node x in the graph, the tree must contain exactly once an *instance* description x/P, and it can additionally contain an arbitrary number of additional references to x as a standalone variable (as many times as the node is argument to a re-entrant role). While both cases have distinct denotations, these denotations are all of the same *type*, namely that of a generalized quantifier $\langle\langle e,t\rangle,t\rangle$.[6] We can thus safely say that a node denotes a set of properties. While this obviously differs from denoting objects or events, it comes very close given that logical representations bear the extra burden of explicitly binding entities to some quantifier and domain of restriction. In the above translation, this binding is built into the denotation of the instance description x/P, which is why x/P does not denote an entity. In contrast, any other reference to the node as a variable x denotes $\lambda f\, f(x)$, which is exactly the standard lifting of an **entity** x to the type $\langle\langle e,t\rangle,t\rangle$. Hence, nodes are seen as plain entity when used as argument to re-entrant roles. In the same order of ideas, note that AMR does not express any distinction between indefinite and definite entities. Thus the AMR in figure 2 is also associated with the phrase "It's the girl who likes herself". To make this reading available, one could for instance non-deterministically switch to a rule like $[\![x/P]\!] = \lambda f\, f(\iota x P(x))$, which,

[5] It offers less control over the relative scoping of quantifiers, but it is sufficient to handle reentrance. Also, it yields semantically equivalent interpretations for all examples discussed by Lai et al. (2020), including donkey sentences.

[6] Assuming that type e is a supertype of all entities, including objects, events and states.

again, makes (x/P) denote a type-lifted entity.[7]

Bos (2016) and Lai et al. (2020) thus clarify the relationship between AMR nodes and entities. Their translations show that, while labeled nodes systematically introduce three distinct components of meaning from a logical perspective (a variable entity, a predicate restricting the domain of this entity, and a binding quantifier), nodes can be thought of as entities, in the sense that the AMR roles linking nodes to one another logically express relations between entities.

3 Copula sentences

Some copula sentences represent a challenge for the view developed in the previous section. AMR guidelines (section *Main verb "be"*), state that copula sentences are represented using the :domain role most of the time. They associate the sentence *The man is a lawyer* with the AMR below:

AMR 1: *The man is a lawyer*

```
(l/lawyer :domain (m/man))
```

Accordingly, we should have the following sentence-AMR association:

AMR 2: *The man who sings is a lawyer*

```
(l/lawyer :domain
  (m/man :arg0-of (s/sing-01)))
```

Applying the translation from section 2 yields the following term:

$$\lambda f\, \exists m\, \mathsf{man}(m) \wedge \exists s\, \mathsf{sing\text{-}01}(s) \wedge \mathsf{arg0}(s,m) \\ \wedge \exists l\, \mathsf{lawyer}(l) \wedge \mathsf{domain}(l,m) \wedge f(l). \qquad (\phi)$$

One of the motivations for relating AMR to logic is to model entailment.[8] Another is to specify the denotation of AMR elements. However, as it stands, ϕ fails at both of these tasks. To see why, consider the sentence *A lawyer sings*, which is entailed by *The man who sings is a lawyer*, and its AMR:

AMR 3: *A lawyer sings*

```
(s/sings :arg0 (l/lawyer))
```

The latter logically translates as:

$$\lambda f\, \exists l\, \mathsf{lawyer}(l) \wedge \exists s\, \mathsf{sing\text{-}01}(s) \wedge \mathsf{arg0}(s,l) \\ \wedge f(s) \qquad (\psi)$$

[7] Moreover, an equivalent treatment of indefinite is likely achievable using Hilbert's epsilon calculus.

[8] Provided of course that we uniformly resolve parts left underspecified by AMR in the putative premise and consequent.

In order to discuss actual propositions rather than sets of continuations, let us define $\mathsf{cls}(\Gamma)$ as $\Gamma(\lambda n\, \top)$ (the closure of Γ under a continuation trivially true of anything). The problem is that $\mathsf{cls}(\psi)$ is not entailed by $\mathsf{cls}(\phi)$, because $\mathsf{lawyer}(m)$ is not entailed by $\mathsf{lawyer}(l) \wedge \mathsf{domain}(l,m)$. Hence, we fail to capture even simple entailments. Comparing ϕ and ψ also leaves us unsure about the denotation of l (and, consequently, lawyer) in these examples. Does it denote a person in both, or a person in the first and a state/property in the second?

Importantly, we are not trying to decide whether copula constructions (or other kinds of predications) systematically introduce states (on this matter, see for instance Asher, 1993; Maienborn, 2005), only whether we can interpret an element like (l/lawyer) **consistently** accross different AMRs assuming either option. If Lai et al. (2020)'s semantics is to achieve this, then we should be able to explain the link between ϕ and ψ without making distinct assumptions about the domain of quantification for l, or the denotation of lawyer in interpreting one or the other.

One way to satisfy this requirement, and solve the entailment issue, is to assume that l denotes a person in both AMRs and interpret the :domain as equating two entities. This makes $\mathsf{lawyer}(l) \wedge \mathsf{domain}(l,m)$ equivalent to $\mathsf{lawyer}(l) \wedge l = m$ which entails $\mathsf{lawyer}(m)$. Unfortunately, this solution is inconsistent with cases of copula constructions with adjectives, because the latter are also handled with :domain. Let us illustrate this with another sentence from the guidelines: *The marble is small*. The annotated AMR is (s/small :domain (m/marble)). If :domain expresses equality of entities, then the logical translation (after closure) of this AMR is equivalent to:

$$\exists m\, \mathsf{marble}(m) \wedge \exists s\, \mathsf{small}(s) \wedge s = m$$

which we can informally paraphrase as: *The marble = a small thing*. This analysis seems dubious. If the sentence hid quantification over the domain of the adjective, one should expect semantic ambiguities, which are not observed for adjective copula constructions. For instance, *Lila believes that the marble is small* does not have the *de re* reading *There is a small thing which Lila believes to be = to the marble*. But crucially, we can also reject it from more AMR-centered considerations. For instance, the AMR for *The marble is very small* is:

```
(s/small :domain (m/marble)
         :degree (v/very)).
```

Interpreting :domain as equality between entities in the latter is nonsensical, since it would let *very* modify an object, and would therefore amount to reading the sentence as *The marble = a small thing which is very*.

We thus reject an interpretation of :domain as equality on the basis that its two arguments are generally of incompatible *types*. In the example above, *(m/marble)* denotes an object, but *(s/small)* must denote something of a more abstract nature, which supports degree modification. Potential candidates for the denotation of *(s/small)* are the property of being small, or a neo-Davidsonian entity representing a "state" of having this property. This is backed up by cases of copula sentences which are not handled with :domain in AMR. We provide two examples below, respectively taken from the guidelines and proxy files of the AMR corpus (Knight et al., 2021), and suitably truncated for the sake of space ([...] indicates truncated roles):

AMR 4: *The boy is a hard worker* (isi_0001.25)

```
(w/work-01 :arg0 (b/boy)
           :manner (h/hard-02))
```

AMR 5: *Iftikhar Ahmed is a Pakistani interior ministry official* (PROXY_AFP_ENG_20020115_0320.13)

```
(p2 / person [...]
  :arg0-of (h/have-org-role-91
             :arg1 (m/ministry [...])
             :arg2 (o/official)))
```

There is arguably no reason to think that these two sentences would relate different types of objects from the ":domain" kind of copula sentences above. AMR 4 relates a person to a work that he performs, and AMR 5 relates a person to an institutional position that he occupies.[9] In both cases, a concrete object is related to a more abstract object akin to a property that the former can have, or a state that it can be in. We take this as evidence that :domain should behave similarly and relate an entity to some property or state.

If these conclusions are correct, then Lai et al. (2020)'s proposal cannot consistently handle AMR representations of copula constructions like AMR 2, because they switch the denotation of an element like $(l/lawyer)$ to a type of object (a property or a state) different from their "standard" denotation in other AMRs like AMR 3. We will now show how to amend the compositional interpretation rules to resolve this inconsistency.

[9] We think that the focus on $p2$ is an annotation mistake and should rather be on h, but this does not change our point.

4 Default entity decomposition

In the previous section, we have discussed a problem with the denotation of *(l/lawyer)* in sentences with the noun *lawyer*. This problem only generalizes to nouns giving rise to :domain edges in copula constructions. It does not occur with nouns that invoke Propbank frames, such as *worker*, or AMR role frames such as *president*. Consider the two AMRs below:

AMR 6: *The man who sings is a worker*

```
(work-01
    :arg0 (m/man :arg0-of (sing-01)))
```

AMR 7: *The worker sings*

```
(s/sing-01
    :arg0 (p/person
            :arg0-of (work-01)))
```

The (closure) of their logical interpretation is given below, respective of the order:

$$\begin{aligned}&\exists m\,\mathsf{man}(m) \wedge \exists s\,\mathsf{sing\text{-}01}(s) \wedge \mathsf{arg0}(s,m)\\ &\qquad\wedge \exists w\,\mathsf{work\text{-}01}(w) \wedge \mathsf{arg0}(w,m)\\ &\exists p\,\mathsf{person}(p) \wedge \exists w\,\mathsf{work\text{-}01}(w) \wedge \mathsf{arg0}(w,p)\\ &\qquad\wedge \exists s\,\mathsf{sing\text{-}01}(s) \wedge \mathsf{arg0}(s,p)\end{aligned}$$

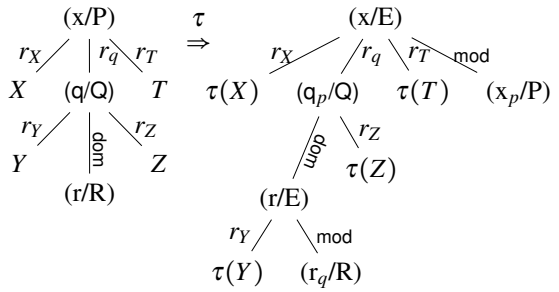
Assuming $\vDash \forall x\, man(x) \rightarrow person(x)$ as a meaning postulate, we predict the entailment from AMR 6 to AMR 7 without further difficulties.

Surely, the difference between *worker* and *lawyer* does not stem from a difference between :arg0 and :domain. Rather, it stems from a difference of focus (in the AMR sense). In the *worker* case, the focus of AMR 6 (*(w/work-01)*) is not an instance of the same concept as the focus of the arg0 role of *(s/sing-01)* in AMR 7. In contrast, in the lawyer case, the focus of AMR 2 and the focus of the arg0 role of *(s/sing-01)* in AMR 3 are instances of the same concept. The ability of the word *worker* to invoke two concepts, a person and a "working", solves the issue because both concepts can claim the focus depending on the use of *worker* in a sentence. In its "standard" use, the focus is on the person, but when used as object of a copula, the focus switches to the "working". This is not possible for *lawyer* basically because there is no PropBank frame corresponding to the activity of "lawing", simply because there is no such word in English.

In order to offer a consistent treatment for all copula sentences, we should therefore try to unify

the treatment of "lucky" words which decompose along some PropBank frame, and "unlucky" ones which do not. An appealing idea to this effect is to provide a default decomposition for every node, *e.g.*, consider (x/P) as syntactic sugar for (x/E :mod (x_p/P)), where E is a vacuous "entity" predicate. However, two obstacles are in the way: 1) without further restrictions, node decomposition would yield infinite AMRs through recursive rewriting. For instance (x/P) would be understood as syntactic sugar for (x/E :mod (x_p/P)), which itself would rewrite as (x/E :mod (x_p/E :mod (x_{p_p}/P))) and so forth and so on. 2) applying the decomposition rule to **both** occurences of (l/lawyer), in 3 and 2 respectively, would leave us with the very same problem regarding the type of (l_p/lawyer) instead of (l_p/lawyer).

Essentially, the solution to both problems is to forbid decomposing the origin of a :domain role (or the target of a :mod role), in order to implement the focus-switching mechanism described above. The idea is to let 'normal' nodes decompose into an AMR (x/E :mod (x_p/P)) focusing the fresh entity node, while leaving nodes with a domain role unaltered and thus keep focus on the original node in the latter case. We let τ denote the resulting "default decomposition" transformation. τ decomposes every node of an AMR tree into an entity modified by a predicate, except when it has a :domain role. It is informally schematized below:



For instance, in AMR 2 both (m/man) and $(s/sing-01)$ receive a default decomposition into an entity (m for *man*, s for *sing-01*) and a state (m_p for being a man, s_p for being a singing), but $(l/lawyer)$ does **not**, because it has a domain edge. The result is the AMR below:

```
(l_p/lawyer
  :domain (m/E :mod (m_p/man)
               :arg0-of (s/E :mod
                    (s_p/sing-01))))
```

In fact, altering the original AMR is not even necessary, since we can directly implement the default decomposition into the denotational semantics. To this extent, let us assume that $V_p = \{x_p, y_p, \dots\}$ is a set of variable symbols disjoint from the one used in AMR trees, such that each AMR variable x can be injectively mapped to a variable x_p in V_p. We introduce a second interpretation function $[\![\cdot]\!]^D$ which is triggered for nodes with an outgoing :domain edge. In all of the following, r ranges over all roles except domain and mod. We abbreviate domain as dom. The "standard" interpretation rules are:

$$[\![c]\!] = \lambda f\, f(c)$$
$$[\![x]\!] = \lambda f\, f(x)$$
$$[\![(\mathbf{N})]\!] = [\![\mathbf{N}]\!]$$
$$[\![x/P]\!] = \lambda f \exists x \exists x_P\, P(x_P) \wedge \mathsf{dom}(x_p, x) \wedge f(x)$$
$$[\![\mathbf{N}\!:\!r\,\mathbf{A}]\!] = \lambda f\, [\![\mathbf{A}]\!](\lambda m\, [\![\mathbf{N}]\!](\lambda n\, r(n,m) \wedge f(n)))$$
$$[\![\mathbf{N}\!:\!r^{-1}\,\mathbf{A}]\!] = \lambda f\, [\![\mathbf{N}]\!](\lambda n\, [\![\mathbf{A}]\!](\lambda m\, r(m,n) \wedge f(n)))$$
$$[\![\mathbf{N}\!:\!\mathsf{dom}\,\mathbf{A}]\!] = \lambda f\, [\![\mathbf{A}]\!](\lambda m\, [\![\mathbf{N}]\!]^D(m)(\lambda n\, \mathsf{dom}(n,m) \wedge f(n)))$$
$$[\![\mathbf{N}\!:\!\mathsf{mod}\,\mathbf{A}]\!] = \lambda f\, [\![\mathbf{N}]\!](\lambda n\, [\![\mathbf{A}]\!]^D(n)(\lambda m\, \mathsf{dom}(m,n) \wedge f(n)))$$

and for nodes with a :domain role:

$$[\![c]\!]^D = \lambda e \lambda f\, f(c)$$
$$[\![x]\!]^D = \lambda e \lambda f\, f(x_p)$$
$$[\![(\mathbf{N})]\!]^D = [\![\mathbf{N}]\!]^D$$
$$[\![x/P]\!]^D = \lambda e \lambda f \exists x_p\, P(x_p) \wedge f(x_p)$$
$$[\![\mathbf{N}\!:\!r\,\mathbf{A}]\!]^D = \lambda e \lambda f\, [\![\mathbf{A}]\!](\lambda m\, [\![\mathbf{N}]\!]^D(e)(\lambda n\, r(e,m) \wedge f(n)))$$
$$[\![\mathbf{N}\!:\!r^{-1}\,\mathbf{A}]\!]^D = \lambda e \lambda f\, [\![\mathbf{N}]\!]^D(e)(\lambda n\, [\![\mathbf{A}]\!](\lambda m\, r(m,e) \wedge f(n)))$$
$$[\![\mathbf{N}\!:\!\mathsf{dom}\,\mathbf{A}]\!]^D = \lambda e \lambda f\, [\![\mathbf{A}]\!](\lambda m\, [\![\mathbf{N}]\!]^D(e)(\lambda n\, \mathsf{dom}(e,m) \wedge f(n)))$$
$$[\![\mathbf{N}\!:\!\mathsf{mod}\,\mathbf{A}]\!]^D = \lambda e \lambda f\, [\![\mathbf{N}]\!]^D(e)(\lambda n\, [\![\mathbf{A}^D]\!](e)(\lambda m\, \mathsf{dom}(m,e) \wedge f(n)))$$

Figure 5 shows how the semantics unpacks for AMR 2. One easily verifies that the result entails AMR 3: the latter interprets as $\exists l \exists l_p\, \mathsf{lawyer}(l_p) \wedge \mathsf{dom}(l_p, l) \wedge \exists s \exists s_p\, \mathsf{sing\text{-}01}(s_p) \wedge \mathsf{dom}(s_p, s) \wedge \mathsf{arg0}(s, l)$ (since 3 does not have any :domain role, each of its nodes is decomposed). To check the entailment, notice that, up to renaming of the quantified variable l to m, every conjunct in the formula above also appears as a conjuct of the interpretation of AMR 2 in figure 5.

To conclude this section, let us discuss some of the properties, benefits and limitations of the proposed default entity decomposition approach. While the target logical interpretations are undoubtedly less readable, they are obtained from the (unaltered) original AMR. So what we have achieved is an improved notion of entailment between AMRs,

$$
\begin{aligned}
&[\![\text{(m/man :arg0-of (s/sing-01))}]\!] \\
&= \lambda f \exists m \exists m_p \, \mathsf{man}(m_p) \wedge \mathsf{dom}(m_p, m) \\
&\qquad \wedge \exists s \exists s_p \mathsf{sing\text{-}01}(s_p) \wedge \mathsf{dom}(s_p, s) \\
&\qquad \wedge \mathsf{arg0}(s, m) \wedge f(m) \\
&[\![\text{l/lawyer}]\!]^D = \lambda e \lambda f \exists l_p \, \mathsf{lawyer}(l_p) \wedge f(l_p) \\
&[\![\text{(l/lawyer :domain (m/ man :arg0-of (s/sing-01)))}]\!] \\
&= \lambda f \, [\![\text{(m/man :arg0-of (s/sing-01))}]\!](\\
&\qquad \lambda m \, [\![\text{l/lawyer}]\!]^D(m)(\lambda n \, \mathsf{dom}(n, m) \wedge f(n))) \\
&= \lambda f \exists m \exists m_p \, \mathsf{man}(m_p) \wedge \mathsf{dom}(m_p, m) \\
&\qquad \wedge \exists s \exists s_p \mathsf{sing\text{-}01}(s_p) \wedge \mathsf{dom}(s_p, s) \\
&\qquad \wedge \mathsf{arg0}(s, m) \\
&\qquad \wedge \exists l_p \, \mathsf{lawyer}(l_p) \wedge \mathsf{dom}(l_p, m) \wedge f(l_p)
\end{aligned}
$$

Figure 5: Interpretation of AMR 2 with default entity decomposition

and a better understanding on the denotation of AMR nodes in copula sentences, at no cost for the annotation capabilities of the formalism.

The attentive reader might have noticed that nodes with a :domain role have an interpretation of type $\langle e, \langle e, t \rangle, t \rangle$ differing from "regular" nodes whose interpretation is of type $\langle \langle e, t \rangle, t \rangle$. While the additional entity-type argument might seem vacuous at first, it is in fact essential to handle (rather frequent) cases of nominal copula constructions where the noun following the copula is itself modified. Consider, as an exemple, the following sentence from the AMR corpus: *Teikovo is a small town in the Ivanovo region about 250 kilometers or 155 miles northeast of Moscow.* The full AMR is given in annex. For our present purpose, we only need to consider the following partial AMR: (t/town :mod (s/small) :domain (c2/city)). Our semantics, ensures that in this context the subtree (t/town :mod (s/small)) denotes the complex property of being a town which is small, and that, as a result, Teikovo (the city) is attributed both properties of being a town and being small. Importantly, the property of being small is not attributed to the *predicate* town, but to the same entity that the latter ends up attributed to, whichever it might be.

Put another way, our semantics implements an intersective treatment of chains of modifiers. Hence, (c/cat :mod (r/grey :mod (f/fierce))) denotes a cat which is both grey and fierce. Of course, adjectival modification is not always intersective, and roles such as degree or time will require a separate treatment. However, the intersective semantics appears necessary to reconciliate some observed variations in annotations, as displayed by the examples below.

The revised interpretation can help us assess some difficult annotation choices. Consider the two sentences *Only if Ron Paul doesn't become president then there will be war* and *And, that is something needed to become President*. Both sentences are from the AMR corpus, and their AMRs are provided in annex. Annotators have made different choices for these two sentences: the first involves (b/become–01 :arg2 (p2/president)) whereas the second involves (b/become–01 :arg2 (p2/person :arg1–of (h/have–org–role–91 :arg2 (p3/president)))). Are these two treatments of *become president* equivalent or incompatible? We can answer this question, at least if we admit that *becomes* is akin to a kind of copula construction[10] and that have-org-role-91 is also akin to :domain in that respect; it is the (reified) relation used to express *e.g.*, *Ron Paul is president*.[11] Under these assumptions, the question amounts to spotting the difference (if any) between (p/president :domain (r/Ron_Paul)) and (x/person :mod (p/president) :domain (r/Ron_Paul)) (ignoring AMR decomposition of named entity, for simplicity). Our semantics associates the former with the proposition $\exists r \exists r_p \, \mathsf{Ron\_Paul}(r_p) \wedge \mathsf{dom}(r_p, r) \wedge \exists p_p \, \mathsf{president}(p_p) \wedge \mathsf{dom}(p_p, r)$ and the latter with the proposition $\exists r \exists r_p \, \mathsf{Ron\_Paul}(r_p) \wedge \mathsf{dom}(r_p, r) \wedge \exists x_p \mathsf{person}(x_p) \wedge \exists p_p \mathsf{president}(p_p) \wedge \mathsf{dom}(p_p, r) \wedge \mathsf{dom}(x_p, r)$. If a president must always be a person, then the two are clearly logically equivalent.

The approach, however, yields arguably puzzling scoping when *e.g.* attitude verbs have copula sentences as objects. While the two AMRs discussed in the previous paragraph have equivalent **closures**, their different focus yield different propositions when embedded. Consider, under the same modeling hypotheses, the sentence *I believe that Ron Paul is president*. The AMR

```
(believe-01 :arg0 (i/I)
            :arg1 (p/president :domain
                  (r/Ron_Paul)))
```

[10] *X becomes Y* is commonly seen as *asserting* that *X* is *Y* and *presupposing* that *X* was not *Y* before. In this view, *XbecomesY* can be thought of as a paraphrase of *X* starts to be *Y*.

[11] become-01 and have-org-role-91 are similar to domain at least regarding the type of objects that they relate, and dealing with them as such would require that we extends our semantics to account for this fact.

could arguably be paraphrased as *I believe that a state of being president obtains which has Ron Paul as theme*, whereas

```
(believe-01 :arg0 (i/I)
  :arg1 (x/person
          :mod (p/president)
          :domain (r/Ron_Paul)))
```

would rather be paraphrased as *Ron Paul is president, and I believe that a state obtains of being a person which has Ron Paul as theme*. We do not commit on whether this is a bug or a feature of the approach. However, we note that we have voluntarily stuck with an approach producing pure first-order target Davidsonian propositions. To render the the two AMRs above fully equivalent (if deemed desirable), one probably needs to allow roles' participants to be higher-order objects like propositions, because as things stand, we are able to express states of 'being a person' or 'being president', but not of 'being a person who is president'.

5 Quasi-predicates

We now turn to a different problem, which originates from the same discrepancy between the words of english which invoke PropBank frames, and those which do not.

The phenomenon at stake is the treatment of what Meaning-Text Theory (MTT) calls quasi-predicates. Mel'čuk and Polguère (2008); Polguère (2012) define predicates as those lexical meanings that have two properties: 1) they denote situations (in a broad sense including events and facts), and 2) they involve a number of semantic participants whose value is contingent, but whose participation is necessary. Polguère notes that predicate are commonly put in opposition to *semantic names*, like the meanings 'rock' or 'star'. Semantic names denote entities rather than situations and they can be defined without reference to participants. Yet, Polguère observes that there is a vast class of meanings, like those of *brother*, *consumer* or *therapist* which denote entities (like semantic names), but cannot denote without accounting for a certain number of participants, due to the presence of a predicate with unbound arguments in their semantic decomposition. These meanings are called quasi-predicates (henceforth, QP).

In essence, a QP can be assimilated to a pair made of an entity and a defining predicate, with focus generally put on the entity (though Polguère (2012) remarks that this is challenged in some constructions, typically copula). For instance, the meaning of *brother* is an entity *X* (a man), combined with a predicate (*X* having a common parent with *Y*). Sometimes, the structure of a QP is displayed explicitly in AMR, *i.e.*, the entity and the defining predicate give rise to different nodes. Consider for instance the example below:

AMR 8: *My brother*

```
(p/person
  :arg0-of (h/have-rel-role-91
   :arg1 (i/I)
   :arg2 (b/brother)))))
```

But this is not always the case:

AMR 9: *Our clients*

```
(c/client :poss (w/we))
```

In this case, the entity *client* itself is linked to the other participant, there is no separation between entity and defining predicate.

While in and of itself these differences are not a problem, they have no other basis than the peculiarities of the English lexicon. The verb *to client* does not exist, and consequently, there is no PropBank frame to decompose the meaning of *client*. The asymmetry between *client* and *brother* is therefore the same as noted in the previous section between *worker* and *lawyer*. What is new however, is that AMR suffers expressive limitations when representing QPs because of this. For instance, we cannot represent the meaning of *The therapist thanks his client* in a way that makes explicit both the fact that the client is the client of the therapist, and that the therapist is the therapist of the client, for we would end up with the cyclic AMR below.

```
(t/thanks-01 :arg0 (t2/therapist :poss
    (c/client :poss t2)) :arg1 c)
```

Interestingly however, reifying (Banarescu et al., 2019) :poss with own-01 removes the cycle:

AMR 10: Reified :poss

```
(t/thank-01
 :arg0 (t2/therapist
  :arg0-of (o1/own-01
   :arg1 (c/client
    :arg0-of (o2/own-01 :arg1 t2))))
  :arg1 c)
```

Reification is introduced in AMR as a mechanism needed to focus a role. But in this case, it is not what it achieves. Rather, reification "borrows" the frame own-01 and uses it as a default to decompose *therapist* and *client*. Indeed, the problem disappears for words which invoke PropBank frames, for instance *The seller thanks the buyer* has an AMR isomorphic to AMR 10.

The default decomposition approach from previous section generalizes this solution, by systematically providing an additional abstract attachment point (the *predicate*) for any outgoing role of the defining predicate of a QP:

AMR 11: Default decomposition

```
(t/E :mod (t_p/thank-01)
 :arg0 (t2/E
  :mod (t2_p/therapist
   :arg1 (c/E
    :mod (c_p/client :arg1 t2)))):arg1 c)
```

Importantly, the approach allows to handle any kind of QP, even if the relationship to their participants is not expressible as a reifiable non-core role like :poss.

6 Conclusion

Whereas seeing AMR nodes as Davidsonian entities seems overall very sound from a logical perspective, we have shown that copula sentences pose an important challenge to this view. The challenge arises because they require to isolate the predicative part of a node from the entity it denotes. We have proposed a unifying mechanism of default decomposition, which systematically entangles the two notions. We have implemented it in a denotational semantics for AMR which does not require any addition to the original AMR annotation. We have shown that this approach generally resolves linguistically unjustified asymmetries depending on the existence of PropBank frames, in particular regarding the representation of quasi-predicates.

References

Nicholas Asher. 1993. *Reference to Abstract Objects in Discourse*. Dordrecht, Boston, and London: Kluwer.

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. In *LAW@ACL*, pages 178–186. The Association for Computer Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2019. Abstract meaning representation (AMR) 1.2.6 specification. https://github.com/amrisi/amr-guidelines/blob/master/amr.md#focus-1. Accessed 2022-11-12.

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

Donald Davidson. 1967. Truth and meaning. *Synthese*, 17(1):304–323.

Kevin Knight, Bianca Badarau, Laura Baranescu, Claire Bonial, Madalina Bardocz, Kira Griffitt, Ulf Hermjakob, Daniel Marcu, Martha Palmer, Tim O'Gorman, and Nathan Schneider. 2021. Abstract Meaning Representation (AMR) Annotation Release 3.0.

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.

Claudia Maienborn. 2005. On the limits of the davidsonian approach: The case of copula sentences. *Theoretical Linguistics*, 31:275–316.

William C. Mann. 1983. An overview of the penman text generation system. In *Proceedings of the Third AAAI Conference on Artificial Intelligence*, AAAI'83, page 261–265. AAAI Press.

Igor A. Mel'čuk. 1973. Towards a linguistic "Meaning-Text" model. In Ferenc Kiefer, editor, *Trends in Soviet Theoretical Linguistics*, pages 33–57. Reidel, Dordrecht.

Igor A. Mel'čuk. 2016. *Language: From Meaning to Text*. Ars Rossica, Moscow/Boston.

Igor A. Mel'čuk and Alain Polguère. 2008. Prédicats et quasi-prédicats sémantiques dans une perspective lexicographique. *Revue de linguistique et de didactique des langues*, 37:99–114.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Terrence Parsons. 1990. *Events in the semantics of English: a study in subatomic semantics*. MIT Press, Cambridge, MA / London.

Alain Polguère. 2012. Propriétés sémantiques et combinatoires des quasi-prédicats sémantiques. *SCOLIA*, 26:131–152.

7 Annex

We reproduce below the annotations for sentences *Only if Ron Paul doesn't become president then there will be war* and *And, that is something needed to become President* discussed in section 4.

```
# ::id DF-200-192451-579_6417.3
# ::tok Only if Ron Paul doesnt become
    president then there will be war .
(w / war-01~e.11
    :condition~e.1 (b / become-01~e.5
        :polarity -
        :ARG1 (p / person :wiki "Ron_Paul"
            :name (n / name
                :op1 "Ron"~e.2
                :op2 "Paul"~e.3))
        :ARG2 (p2 / president~e.6)
        :mod (o / only~e.0))
    :time (t / then~e.7))



# ::id bolt-eng-DF-170-181103-8886306_0011.6
# ::tok And , that is something needed to
    become President .
(a / and~e.0
    :op2 (n / need-01~e.5
        :ARG0 (b / become-01~e.7
            :ARG2 (p2 / person
                :ARG1-of (h / have-org-role-91
                :ARG2 (p3 / president~e.8))))
        :ARG1 (s / something~e.4)))



# ::id PROXY_APW_ENG_20080515_0931.17
# ::snt Teikovo is a small town in the Ivanovo
    region about 250 kilometers or 155 miles
    northeast of Moscow.
(t / town
    :mod (s / small)
    :location (p / province :wiki "Ivanovo"
        :name (n / name :op1 "Ivanovo"))
    :location (r / relative-position
        :op1 (c / city :wiki "Moscow"
            :name (n2 / name :op1 "Moscow"))
        :direction (n3 / northeast)
        :quant (a / about
            :op1 (d / distance-quantity
                :quant 250
                :unit (k / kilometer))))
    :domain (c2 / city :wiki "Teykovo"
        :name (n4 / name :op1 "Teikovo")))
```