Proceedings of the 16th Joint ACL-ISO Workshop Interoperable Semantic Annotation (ISA-16), pages 36–48
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

Annotation-based Semantics

Kiyong Lee
Korea University
Seoul, Korea
ikiyong@gmail.com

Abstract

This paper proposes a semantics *ABS* for the model-theoretic interpretation of annotation structures. It provides a language *ABSr* that represents *semantic forms* in a (possibly λ-free) type-theoretic first-order logic. For semantic compositionality, the representation language introduces two operators $\oplus$ and $\oslash$ with some subtypes for the conjunctive or distributive composition of semantic forms. *ABS* also introduces a small set of logical predicates to represent semantic forms in a simplified format. The use of *ABSr* is illustrated with some annotation structures that conform to ISO 24617 standards on semantic annotation such as ISO-TimeML and ISO-Space.

Keywords: annotation structure, semantic forms, logical predicates, conjunctive or disjunctive composition

1. Introduction

This paper has two aims: [i] to formulate a semantics, called *Annotation-based Semantics* (*ABS*), for the model-theoretic interpretation of annotation structures and [ii] to recommend it as a semantics for ISO 24617 standards on semantic annotation frameworks such as ISO-TimeML (ISO, 2020) or ISO-Space (ISO, 2020). As a semantics for these annotation frameworks, *ABS* has two roles. One role is to validate the abstract syntax that formally defines each annotation framework in set theoretic terms (Bunt, 2010). The other is to interpret the annotation structures that are generated by, or conform to, a relevant annotation framework (see (Lee, 2018) and (Pustejovsky et al., 2019)).

ABS is a structurally simple semantics, consisting of [i] a representation language *ABSr* and [ii] a finite set of logical predicates that are used in *ABSr*, but are defined as part of a model structure like meaning postulates or word meanings as introduced by Carnap (1947 1956) and Montague (1974), as shown in Figure 1, and further developed by Dowty (1979) and Pustejovsky (1995).

The rest of the paper develops as follows: Section 2 provides some motivations for *ABS*. Section 3 describes the basic design of *ABS*. Section 4 defines the type-theoretic first-order predicate logic-based representation language *ABSr*. Section 5 breifly outlines some characteristics of an interpretation model structure for *ABS*. Section 6 shows how the composition rules of *ABSr* apply to the annotation structures that conform to some of the ISO 24617 standards on semantic annotation. Section 7 introduces some related works and discuses the convertibility of semantic forms of *ABS* to DRSs or λ-formulas. Section 8 makes some concluding remarks.

2. Motivation for *ABS*

The main motivation of *ABS* is to lighten the burden of automatically generating intermediary interpretations, called *semantic forms* or *logical forms*, of semantic annotation structures for both human and machine learning or understanding. For this purpose, *ABS* and its representation language *ABSr* introduce two minor operational modifications into the two well-established and model-theoretically interpretable representation languages, the type-theoretic λ-calculus, used for Montague Semantics (MS) (Montague, 1974), and Kamp and Reyle (1993)'s Discourse Representation Theory (DRT). The representation language *ABSr* of *ABS* is designed to to be free from λ-operations, especially involving higher-order variables, by replacing the operation of substitution through the λ-conversion with an equation solving approach (see Lee (1983)), or to convert its semantic forms into visually more readable Discourse Representation Structures (DRSs) preferably without introducing embedded or stacked structures into them. From a theoretical point of view, neither *ABS* nor *ABSr* is totally different from Bunt (2020b) or his earlier efforts to develop an annotation-based semantics with the interpretation function I to convert or annotation structures, defined in abstract (set-theoretic) terms, to DRSs based on Kamp and Reyle (1993)'s Discourse Representation Theory (DRT). From a practical point of view, *ABS* is characterized by dividing the task of interpreting annotation structures between the representation of simpler or *abbreviated* semantic forms and their interpretations enriched with lexical meaning in the form of meaning postulates that constrain the set of possible interpretation model structures.

Based on a type-theoretic first-order predicate logic (*FOL*), *ABSr* is augmented with [i] a small set of operators and [ii] a set of logical predicates. As is developed in Section 3, for any **a** that refers to the abstract specification of an annotation structure or its substructures, either an entity or a link structure, preferably through its ID, the operator σ maps **a** to a semantic form σ(**a**), represented in a first-order logic, while the two non-Boolean operators $\oplus$ and $\oslash$, with their finer-grained subtypes of *merging*, each relate σ(**a**) to another semantic form, constrained by their semantic type. Without much depending on the particular syntactic analysis of each input, these operators combine, in a compositional manner, the pieces of information conveyed by each annotation structure or its substructures into a model-theoretically interpretable logical form, called *semantic form*, in *FOL*. Besides the Boolean connectives in *FOL*, these non-Boolean operators are needed to combine semantic forms that are not of type t (sentential type) as bridges that connect annotation structures to logical forms: for instance, to combine $\sigma(Fido)$ of individual entity type e with $\sigma([runs(e) \wedge agent(e,x)])$ of type $e \rightarrow (v \rightarrow t)$ without using λ-operations in an overt way.

As is elaborated in Section 3, ABS also introduces a small set of *logical predicates* into its representation language *ABSr* and treats them as meaning postulates that constrain a model structure (see Montague (1974) and Dowty (1979)). There are at least two reasons for the introduction of a small set of logical predicates. One reason is *representational simplicity*: it can, for instance, represent the semantic form of the past tense of a verb in English as $\mathbf{past}(e)$, where **past** is a predicate to be defined as part of an interpretation model and e is a variable of type v for eventualities, instead of introducing one of its definitions, which is the most common one $[\tau(e) \subseteq t \wedge t \prec n]$ into the semantic form. This semantic form requires the introduction of a real-time function τ from events to times, two temporal relations, those of inclusion $\subseteq$ and precedence $\prec$, and the notion of the present time n. Furthermore, it is a straightforward process to translate an entity structure like **event**(e1, ran, *pred*:run, *tense*:past) into a semantic form $[run(e_1) \wedge \mathbf{past}(e_1)]$.
Another reason is *representational flexibility*. *ABS* can first choose an appropriate definition or meaning from a set of possible definitions given in a model structure and then decide on an appropriate model M and an assignment g that together satisfy a semantic form like $[run(e_1) \wedge \mathbf{past}(e_1)]$. This would be the case particularly if the past tense needs to be interpreted in a deitic or situational sense, as discussed by Partee (1973) and Quirk et al. (1985).
ABS upholds the principle of minimalism and partiality in its representation. It does not aim nor claim to treat the total interpretation of natural language expressions. Being based on a restricted set of markables in data, either textual or audio-visual, and their annotation, the task of annotation and that of its semantics such as *ABS* are bound to be restrictive: the semantics can be either simple or complex depending on what needs to be annotated. The granularity or complexity of semantic forms only depends on that of the input annotation structures and their substructures. The granularity of perceiving and constructing these structures, especially involving spatio-temporal information, is controlled or modulated through common-sense logic by the need of their applications, as is discussed by Miller and Shanahan (1999) and Gordon and Hobbs (2017)).

3. Basic Design

3.1. Basic Assumptions

The main characteristics of *ABS* are the following. First, *ABS* is based on annotation work, making use of the semantic annotation of coumminicative linguistic data for their semantic interpretation. Without relying on a pre-defined syntax, it manipulates minimally what is encoded in annotation structures and their substructures and converts these structures to logical forms that can be interpreted model-theoretically. *ABS* is, for instance, designed to support spatio-temporal annotation by validating the abstract syntax of ISO-Space (ISO, 2020), as proposed and outlined by Lee (2016), Lee (2018), and Lee et al. (2018) as well as ISO-TimeML (ISO, 2012) and Pustejovsky et al. (2010).
Second, *ABS* only provides *partial* information on a restricted set of markables for semantic annotation. Unlike ordinary semantics like Montague Semantics (Montague, 1974) or even Minimal Recursion Semantics (Copestake et al., 2005), *ABS* is not a general semantics that attempts to treat all aspects of language in an abstract way.
Third, *ABS* leaves much of the information *unspecified*. It allows, for instance, some variables to occur unbound in well-formed semantic forms, as in the interval temporal logic of Pratt-Hartmann (2007), while their scoping is left unspecified till the last stage of composing semantic forms or being interpreted (model-theoretically), unless the scope is specified as part of annotation. As a result, the semantic type of semantic forms is partially non-deterministic: it can be interpreted either as of type t potentially denoting a proposition or a truth-value or of a functional type $\alpha \rightarrow t$, where α is a well-defined type, denoting a set of individual objects or of higher-order objects.
Fourth, *ABS* introduces a small set of predicates such as **past** and **perfective** for the specification of tense and aspect. It can also introduce the predicates **holds** and **occurs**, as defined in Allen (1984) and others, for the event-type dependent temporal anchoring into semantic forms. All these predicates that occur in semantic forms are defined as part of an interpretation model or leaving room for various uses of grammatical concepts or their contextually dependent interpretations.
Being based on annotations, *ABS* must deal with complex issues in semantic annotation such as quantification, for instance, as raised by Bunt (2020a) and Bunt (2020b) or the meaning of determiners that include numerals as in "two donkeys" in language in general. It may also have to deal with the structure and substructures of eventualities, especially dealing with dynamic motions, as discussed in Mani and Pustejovsky (2012). The complexity or granularity of *ABS* thus totally depends on that of annotation structures or the type of annotations.
In addition, *ABS* upholds a couple of well-established basic assumptions as its theoretical basis:

1. Semantics is constrained by a type theory (Montague semantics: Montague (1974) and Dowty et al. (1981))
2. Events are viewed as individuals (Neo-Davidsonian semantics: Davidson (1979), Davidson (2001), Parsons (1990), Pustejovsky (1995))
3. Variables are linked to discourse referents (Discourse representation theory: Kamp and Reyle (1993))

3.2. Metamodel

Figure 1 shows the general design of *ABS* , which consists of (1) a representation language *ABS* and (2) an interpretation model M with logical predicates defined.
ABS is an annotation-based semantics, meaning that its representation language *ABSr* translates each **a** of the abstract specification of entity or link structures that constitute annotation structures to a well-defined semantic form $\sigma(\mathbf{a})$. *ABS* then interprets each semantic form $\sigma(\mathbf{a})$ with respect to a model M, a list *D* of definitions of logical predicates, and an assignment g of values to variables, $[\![\sigma(\mathbf{a})]\!]^{M,D,g}$. Each $\sigma(\mathbf{a})$ in *ABSr* is an expression of first-order logic, but each of the logical predicates that my occur in $\sigma(\mathbf{a})$ may be defined in terms of higher-order logic as part of the model structure.

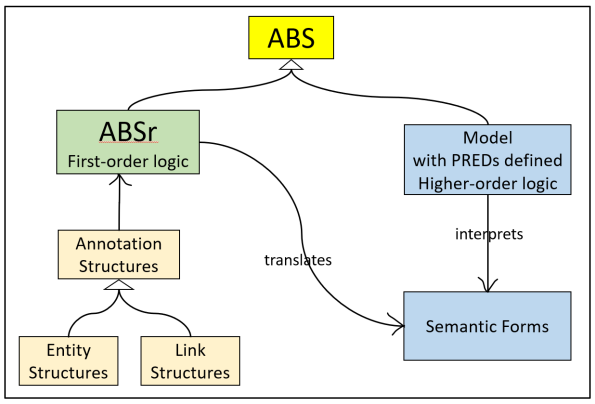

Figure 1: metamodel of *ABS*

4. Syntax

ABSr subsumes a type-theoretic first-order predicate logic (*FOL*). This means: [i] every well-formed (semantic) form in *ABSr* is assigned one of the types, as specified in 4.1, and [ii] every well-formed formula in *FOL* is well-formed and of type t (having a truth-value) in *ABSr*.

4.1. Basic and Functional Types

ABS adopts the system of semantic types which Kracht (2002) and Pustejovsky et al. (2019) have developed. They extend the list of basic types from Montague (1974)'s basic set of types {e, t} to an enlarged list, as specified in (1).

(1) Extended List of Types:
[i] **Basic Types**:
a. t, the type of truth-values
b. e, the type of individual entities
c. v, the type of eventualities
d. i, the type of time points
e. p, the type of spatial points
f. m, the type of measures
g. int, the type of intervals
h. vec, the type of vectors[1]
[ii] **Functional Types**:
h. If α and β are any types, then $\alpha \rightarrow \beta$ is a type.

Type constructors such as $\rightarrow$ are introduced to define *functional types*: e.g., $e \rightarrow t$, $v \rightarrow t$, $i \rightarrow t$, $p \rightarrow t$ or $e \rightarrow (e \rightarrow t)$. Eventuality descriptions such as *run* or *love* are of type $v \rightarrow t$, which is abbreviated to $\mathcal{E}$ (see Pustejovsky (1995)), while the same symbol $\mathcal{E}$ is also used as as a symbol for a variable ranging over a set of eventualities or instances of an eventuality. The functional type $p \rightarrow t$, denoting a set of spatial points, is often represented by a type r of regions[2] I may call these functional types $\mathcal{E}$ and r *pseudo-basic types*, for they are seldom analyzed as functional types.
As introduced by Pustejovsky et al. (2019), path types are defined on the basis of the type of intervals int, which is defined $[0,1] \subset R$, where R is a set of reals. A path π will be that function $int \rightarrow p$, which indexes locations on the path to values from the interval [0,1] (see Pustejovsky et al. (2019)). A vector path π_v can also be defined as $int \rightarrow vec$. An event path π_v will be defined as $v \rightarrow \pi_v$ as the function from eventualities to the vector path.
Kracht (2002) and Pustejovsky et al. (2019) also introduce the group operator $^\bullet$ to form group types, for example, $p^\bullet$ for the group of spatial points. Link (1998) introduces two symbols * and * and prefixes them to a predicate P to generate the group predicate $^*P$ and the plural predicate $^\star P$, both based on the predicate P.
Corresponding to each of the IDs of annotation structures or its substructures, entity or link structures, and of each of the types as defined in (1), there is a list of variables. Some of them are listed below:

Categories[3]	Ids	Types	Variables
annotation	a_1,...	t	$a_1, ...$
entity	x_1,...	e	$x, x_1, ...$
		v	$s, e, e_1, ...$
event	e_1,...	$\mathcal{E}, e \rightarrow t$	$\mathcal{E}, ...$
timex3	t_1,...	$\mathcal{I}, i \rightarrow t$	$t, t_1, ...$
place	pl_1,...	$r, p \rightarrow t$	$l, l_0, ...$
path	p_1,...	$\pi_v, int \rightarrow p$	$p, p_1, ...$
event-path	ep_1,...	π_ϵ	$v \rightarrow \pi_v$
measure	me_1,...	m	$m, m_1, ...$
link	l_1,...	t	

Table 1: IDs, variables, and types

The list of variables is just a conventionally used list. To be precise, for each entity structure E that confirms to a recognized annotation scheme such as ISO-TimeML or ISO-Space, a variable is defined as a pair $<var{:}\tau>$, where var is a variable and τ is a type. Conventionally, any lowercase Latin characters such as x, y, etc. or e and s are used as variable for any one of the basic types provided that its type is specified: for example, $x{:}<var, p \rightarrow t>$ to use x as a variable ranging over regions of type r, or $p \rightarrow t$. Uppercase Latin characters or special characters like $\mathcal{E}$ are used for functional types: $\mathcal{E}$ is a variable for eventuality descriptions such as what is denoted by a verb like "run". Note that $run(e)$ is of type t, while the eventuality description run is of type $v \rightarrow t$ and its argument e is a variable of eventuality type v.[4]

4.2. Syntax Proper

The part of *ABSr* that introduces the *merge* operators and their use is defined by $Syntax_{absR}$. This syntax specifies what constitutes *ABSr* and how its constituents are formed. Some preliminary remarks are made before specifying the syntax of *ABSr*.

4.2.1. Preliminary Remarks

Just like any language, the representation language *ABSr* is a language that consists of a non-empty set of strings of character symbols. Each of such character strings in

[1] (g) and (h) are my own additions to the list of basic types.
[2] See Mani and Pustejovsky (2012) for the discussion of 3.2.2 regions as primitive objects vs. 3.2.3 regions as sets of points.
[4] Here, it is a bit confusing to use e as standing for a basic type for individual entities and use it as referring to an eventuality of type v: e.g. $[run_{v \rightarrow t}(e_v) \wedge \mathbf{agent}(e,x)]_{e \rightarrow (v \rightarrow t)}$.

ABSr is called a *semantic form* because it serves as an intermediary form for the model-theoretic interpretation of annotation structures. Further to clarify what *ABSr* is, I make some technical remarks.

Remark 1: Mapping σ For any **a** that refers to the *abstract specification* of each of the entity or link structures which together constitute an annotation structure, independent of how these structures are represented, σ maps **a** to a semantic form in *ABSr*. $\sigma(\mathbf{a})$ is read as "the semantic form of **a**" in *ABSr* and is a *well-formed form* (wff) of *ABSr*.
$\sigma(\mathbf{a})$ is considered independent of the format that represents it, but has to check the abstract syntax that validates the abstract specification **a**. Hence, **a** must be the same as the interpretation function I that is introduced in Bunt (2020b) and Bunt (2020a).

Remark 2: Model-theoretic Interpretation The symbol [[ ]] is used to represent a (model-theoretic) denotation. Given any semantic form $\sigma(\mathbf{a})$ in *ABSr*, its denotation with respect to a model M, an assignment g of values to variables, and a set D of definitions for **logical predicates** is represented by $[\![\sigma(\mathbf{a})]\!]^{M,g,D}$.

Remark 3: Typing *ABSr* is a type-based language. Hence, every well-formed (semantic) form A and any c of its constituents such as variables in *ABSr* is assigned a type. The type τ of A or c is represented as a pair: e.g., $<A{:}\tau>$, $<c{:}\tau>$, $<var{:}\tau>$, or as a subscript to A or one of its constituents: A_τ, c_τ or x_e.

4.2.2. Formulation of Syntactic Rules

Like the syntax of an ordinary language, $Syntax_{absR}$ consists of a vocabulary and a set of formation rules, as specified in (2).

(2) $Syntax_{absR}$ = $<V,R>$ such that
a. V is a vocabulary that includes binary *merge* operators $\{\oplus, \oslash\}$ over the set of semantic forms in *ABSr* and their subtypes, and
b. R is a set of composition rules for merging, as formulated in (7).

There are two sorts of well-formed semantic forms (swff) in *ABSr*: basic and composed, each defined by a rule in R, a list of rules, in (4.2.3) and (7).

4.2.3. Atomic Semantic Forms

Atomic semantic forms are defined by Rule A.

(3) **Rule A** for Atomic semantic forms:
For any abstract specification $\mathbf{a}_{Ec}$ of an entity structure E of category c,[5] and a type τ associated with *cat*,
$\sigma(\mathbf{a}_{Ec})_\tau$ is a well-formed form of type τ in *ABSr*.

Remark 4: $\mathbf{a}_{Ec}$ in $\sigma(\mathbf{a}_{Ec})_\tau$ is replaced by the ID of Ec.

Following DRT (Kamp and Reyle, 1993), the new occurrences of variables in a semantic form are registered.

(4) **Rule A.1** for Variable Registry:
Any variable that is newly introduced to $\sigma(\mathbf{a}_{Ec})$ is listed in the preamble: i.e., $\Sigma_{var:type}\sigma(\mathbf{a}_{Ec})$.
Note: These variables may not be registered if they can be recognized contextually.

The variables in the preamble $\Sigma_{var:type}$ are treated as *discourse referents*, to which each occurrence of the variables in $\sigma(\mathbf{a}_{Ec})$ is bound.
Consider an example, annotated as in (5):

(5) a. Fido ran_{w2} away_{w3}.

b. Annotation(id=a5)
event(e1, w2-3, *pred*:run, *tense*:past)

c. Semantic form:
$\sigma(e1_e)_\alpha := \{e_1{:}e\}[run(e_1)_t \wedge past(e_1)_t]_\alpha$
where ":=" is a meta-symbol standing for "is".

Some notes are needed here. (1) For now temporally, the type of $\sigma(e1)$ is left unspecified: it is only marked with α, whereas the type of e_1 in the registry is specified as the individual type e. (2) The ID "e1" in $\sigma(e1)$ does not refer to the entity structure of category **event**, but its abstract specification that conforms to the abstract syntax of the relevant annotation scheme. (3) The representation of $\Sigma_{var:type}\sigma(\mathbf{a}_{Ec})$ is exactly the same as DRS except that $\sigma(e1)$ in *ABSr* is typed, as in Bos et al. (2017)'s Groningen Meaning Bank (GMB). The semantic form in (5) can be converted to a type-based DRS except that the type of the entire DRS is not specified.

(6)

$e_1{:}e$
$\text{run}(e_1)_t$
$\text{past}(e_1)_t$

4.2.4. Composed Semantic Forms

The current version of *ABSr* introduces two *merge* operators, $\oplus$ and $\oslash$, and their subtypes each marked with a different superscript to represent the merging of (1) two semantic forms or (2) a pair of semantic forms with a functor-like semantic form. The second type of merging is motivated by the treatment of tripartite link structures of the form $<\eta, E, \rho>$, where ρ is a type of relation between an entity η and a set E of entities, in *ABSr*.
These operators are non-Boolean connectives. They are needed to be able to merge semantic forms of type other than the truth-type t. More operators may need to be introduced to treat finer-grained compositions, especially involving the semantics of determiners that include generalized quantifiers, plurals, and the merging of scopes. As suggested by Bunt (personal communication), different symbols will be introduced to represent various subtypes of composition.[6]
For the formulation of composition rules, it is assumed that these rules hold for any well-formed semantic forms A_α, B_β, and C_γ, each of which is typed as α, β, and γ, respectively. For these semantic forms, there are two major types

[5] In a concrete syntax, this category is often called *tag* or *element*.

[6] Bunt (2020b), for instance, introduces the *scope merge operator* $\oplus^s$ and the *possessive scoped merge operator* $\oplus^{ps}$.

of composition, conjunctive ($\oplus$) and distributive ($\oslash$), and then their subtypes:

(7) Types of composition:
Conjunctive composition ($\oplus$):
Rule 1^{bo} Boolean conjunctive composition ($\oplus^{bo}$)
Rule 1^{fa} Functional conjunctive composition ($\oplus^{fa}$)
Rule 1^{sub} Substitutive conjunctive composition by substitution ($\oplus^{sub}$)
Rule 1^{eq} Equative conjunctive composition by equation solving ($\oplus^{eq}$)
Disjunctive composition ($\oslash$):
Rule 2 Disjunctive composition ($\oslash$)
Rule 2^{int} Intensional disjunctive composition ($\oslash^{int}$)
Rule 2^{imp} Implicational disjunctive composition ($\oslash^{imp}$)

Rule 1^{bo} **Boolean conjunctive composition ($\oplus^{bo}$)** is the most common type of composition, as formulated in

(8) **Rule 1**^{bo}: Boolean conjunctive composition:
a. $[A_t \oplus^{bo} C_t]_\alpha := [A_t \wedge C_t]_t$
b. $[\{A_t, B_t\}_\alpha \oplus^{bo} C_t] := [[A_t \wedge B_t]_t \wedge C_t]$

Rule 1^{bo} applies to most of the annotation structures in ISO-TimeML (ISO, 2012), ISO-Space (ISO, 2020), and ISO standard on semantic role annotation (ISO, 2014). For illustration, consider (9):

(9) a. Fido is barking.

b. Entity Structures:
entity(x1, w1, *type*:dog, *form*:nam)
event(e1, w2-3, *pred*:bark, *tense*:present, *aspect*:progressive)

c. Link Structure:
srlink(e1, x1, agent)

The annotation of text fragment (9a) consists of a list of entity structures in (b) and a link structure (c) over them. Here, **srlink** specifies the semantic role of the participant x_1 as an **agent** participating in the event e_1 of barking, as illustrated in (10).

(10) a. Semantic forms of the entity structures:
$\sigma(x1)_t := \{x_1{:}e\}[dog(x_1) \wedge \mathbf{named}(x_1, Fido)]$
$\sigma(e1)_t := \{e_1{:}v\}[bark(e_1) \wedge \mathbf{presProg}(e_1)]$

b. Semantic form of Semantic role link:
$\sigma(srlink)_t$
$:= \{x_1{:}e, e_1{:}v\}$
$[\{\sigma(x1)_t, \sigma(e_1)_t\} \oplus^{bo} agent(e_1, x_1)_t]$
$:= \{x_1{:}e, e_1{:}v\}$
$[[\sigma(x1)_t \wedge \sigma(e_1)_t] \wedge agent(e_1, x_1)_t]$
$:= \{x_1{:}e, e_1{:}v\}$
$[[dog(x_1) \wedge \mathbf{named}(x_1, Fido)] \wedge$
$[bark(e_1) \wedge \mathbf{presProg}(e_1)] \wedge agent(e_1, x_1)]$

c. Semantic form of annotation structure:
$\sigma(a_9)$
$:= \{x{:}e, e{:}v\}\sigma(srlink)$
by Variable renaming and binding
$:= \{x{:}e, e{:}v\}[bark(e) \wedge \mathbf{presProg}(e)] \wedge$
$\mathbf{agent}(e, x)]$

Rule 1^{fa} **Functional conjunctive composition** reflects the functional application of a functor applying to its argument(s) in Montague Semantics (Montague, 1974) or (Dowty et al., 1981). Rule 1^{fa} is formulated in (11):

(11) Rule 1^{fa} Functional conjunctive composition:
a. $[A_\alpha \oplus^{fa} C_{\alpha\to t)}] := [A_t \wedge C_t]$
or
b. $[\{A_\alpha, B_\beta\}] \oplus^{fa} C_{\beta\to(\alpha\to t)}] := [[A_t \wedge B_t] \wedge C_t]$

Example (9) can be analyzed in terms of a functor-argument analysis by assigning a functional type $\alpha \to t$, where α is a type, to the type of each of the annotation structures.

(12) a. Semantic forms of the entity structures:
$\sigma(x1)_{e\to t}$
$:= \{x_1{:}e\}[dog(x_1) \wedge \mathbf{named}(x_1, Fido)]$
$\sigma(e1)_{v\to t}$
$:= \{e_1{:}v\}[bark(e_1) \wedge \mathbf{presProg}(e_1)]$

b. Semantic form of Semantic role link:
$\sigma(srlink)$
$:= \{x_1{:}e, e_1{:}v\}$
$[\{\sigma(x1)_{e\to t}, \sigma(e_1)_{v\to t}\} \oplus^{fa}$
$\mathbf{agent}(e_1, x_1)_{(v\to t)\to((e\to t)\to t)}]$
$:= \{x_1{:}e, e_1{:}v\}$
$[[\sigma(x1)_t \wedge \sigma(e_1)_t] \wedge \mathbf{agent}(e_1, x_1)_t]$
$:= \{x_1{:}e, e_1{:}v\}$
$[[dog(x_1) \wedge \mathbf{named}(x_1, Fido)]_t$
$\wedge [bark(e_1) \wedge \mathbf{presProg}(e_1)]_t \wedge$
$\mathbf{agent}(e_1, x_1)_t]_t$

c. Semantic form of annotation structure:
$\sigma(a_9)$
$:= \{x{:}e, e{:}v\}\sigma(srlink)$
$:= \{x{:}e, e{:}v\}$
$[[dog(x) \wedge \mathbf{named}(x, Fido)]$
$\wedge [bark(e) \wedge \mathbf{presProg}(e)] \wedge \mathbf{agent}(e, x)]$
by Variable renaming and binding

The functional composition with the operator $\oplus^{fa}$ is equivalent to the functional application in λ-calculus, as shown by (13):

(13) a. Arguments:
$\sigma(x1)_{e\to t}$
$:= \lambda x_1[dog(x_1) \wedge \mathbf{named}(x_1, Fido)]$
$\sigma(e1)_{v\to t}$
$:= \lambda e_1[bark(e_1) \wedge \mathbf{presProg}(e_1)]$

b. Funtor for Semantic role link applying to the two arguments in (a):
$\sigma(srlink)$
$:= [\lambda Q[\lambda P[P(x_1) \wedge Q(e_1) \wedge$
$\mathbf{agent}(e, x)](\sigma(e_1))](\sigma(x_1))]$

By applying four λ-conversions to (13b), we obtain the same result as (12c). One noticeable problem with the functional application in λ-calculus is the placing of the arguments in the right order when the functor applies to them.

Rules 1^{sub} and 1^{eq}, subtypes of conjunctive composition, are needed when one of the inputs to links is treated as of some basic or pseudo basic type. Consider the same example (9) but with a different semantic treatment:[7]

(14) a. $\sigma(x1)_e := fido_e$
$\sigma(e1)_{v\to t}$
$:= \{e_1{:}v\}[bark(e_1)\wedge \mathbf{presProg}(e_1)]$

b. $\sigma(srlink3)$
$:= \{e_1{:}v\}$
$[\{\sigma(x1)_e, \sigma(e1)_{v\to t}\} \oplus^{sub}$
$\mathbf{agent}(e_1, x_1)_{(v\to t)\to(e\to t)}]$
$:= \{e_1{:}v\}$
$[\sigma(e1)_t \wedge \mathbf{agent}(e_1, fido)_t]$
$:= \{e_1{:}v\}$
$[[bark(e_1) \wedge \mathbf{presProg}(e_1)]_t \wedge$
$\mathbf{agent}(e_1, fido)]$

The substitution simply replaces some occurrences of a variable with something like a name *fido*.
The equation solving composition ($\oplus^{eq}$) also deals with basic types like names or measures. There is no substitution, but something like $fido_e$ turns into an equation that does not carry kinds of information other than what is stated, as shown in (15):

(15) a. $\sigma(x1)_e := \{x_1{:}e\}[x_1{=}fido_e]_t$

b. $\sigma(e1)_t$
$:= \{e_1{:}v\}[bark(e_1)\wedge \mathbf{presProg}(e_1)]$

c. $\sigma(srlink4)$
$:= \{x_1{:}e, e_1{:}v\}$
$[\{\sigma(x1)_e, \sigma(e_1)_t\} \oplus^{eq} \mathbf{agent}(e_1, x_1)_t]$
$:= \{x_1{:}e, e_1{:}v\}$
$[[\sigma(x_1) \wedge \sigma(e_1)_t] \wedge \mathbf{agent}(e_1, x_1)_t]$
$:= \{x_1{:}e, e_1{:}v\}$
$[[x_1{=}fido] \wedge [bark(e_1) \wedge \mathbf{presProg}(e_1)]_t \wedge$
$\mathbf{agent}(e_1, x_1)]$

d. $\sigma(a_9) := \sigma(srlink4)$

Now by the rule of substitution of identicals in FOL, we have:

(16) $\{e_1{:}v\}$
$[[bark(e_1) \wedge \mathbf{presProg}(e_1)] \wedge \mathbf{agent}(e_1, fido)]$

Unlike the equation solving approach proposed here, Kamp and Reyle (1993) represents names like Fido as $Fido(x)$ of type t in DRSs. This is acceptable but fails to apply the substitution of identicals. Note also that the equation solving approach can be extended to basic types other than entity type e.

Rule 2 Distributive Composition ($\oslash$):
$[\{A_\alpha, B_\beta\} \oslash C_{\beta\to(\alpha\to t)}] := [A'_t \to_c B'_t]_t$,
where $\to_c$ refers to an implication the type of which needs to be specified for each case and A' and B' are minimal modifications of A and B.

[7] In practice, the semantic treatment of names is much more complicated than treating it merely for its referential use. Kamp and Reyle (1993) treat names like "John" as a predicate, thus representing it as *John(x)* in a DRS.

The conjunctive operator $\oplus$ and its subtypes generate truth-functional conjunctions. In contrast, the distributive operator $\oslash$ possibly with its subtypes generates non-conjunctive relations of implication the type or meaning of which needs further analysis.

4.3. Additional Illustrations

Rule 1^{fa} Functional conjunctive composition with ($\oplus^{fa}$) applies to link structures that relate non-basic type entity structures. Consider example (17)

(17) a. John died$_{w2}$ last$_{w3}$ year$_{w4}$.

b. Annotation (id=a$_{17}$):
Entity structure:
event(e1,w2, *pred*:die, *tense*:past)
timex3(t1,w3-4, *type*:date, *value*:2019-XX-XX)
Link structure:
tlink(e1,t1, isIncluded(e1,t1))

(18) a. Semantic form of entity structures:
$\sigma(e1) := \{e_1\}[die(e_1) \wedge \mathbf{past}(e_1)]$
$\sigma(t1) := \{t_1\}[year(t_1,2019)]$

b. Semantic form of temporal link structure:
$\sigma(tlink)$
$:= \{e_1, t_1\}[\{\sigma(e1)_{v\to t}, \sigma(t1)_{i\to t}\}$
$\oplus^{fa} \mathbf{occurs}(e_1, t_1)_{(i\to t)((v\to t)\to t)}]$
$:= \{e_1, t_1\}[[\sigma(e1)_t \wedge \sigma(t1)_t] \wedge \sigma(tlink)_t]$
$:= \{e_1, t_1\}[[die(e_1) \wedge \mathbf{past}(e_1)] \wedge year(t_1,2019)$
$\wedge \mathbf{occurs}(e_1, t_1)]$

c. Semantic form of annotation structure:
$\sigma(\mathbf{a}_{17})$
$:= \{e, t\}\sigma(tlink)$
$:= \{e, t\}[die(e) \wedge \mathbf{past}(e) \wedge year(t,2019)$
$\wedge \mathbf{occurs}(e, t)]$

Rule 1^{eq} Equation solving ($\oplus^{eq}$) applies to the annotation structures that contain names or other basic types. Consider an example taken from Pustejovsky et al. (2019) that introduce the semantics of ISO-Space.

(19) a. [**Gothenburg**$_{pl1}$] is [**in**$_{s1}$] [**Sweden**$_{pl2}$].
b. [[Gothenburg]] = G, $<G{:}p\to t>$
c. [[Sweden]] = S, $<S{:}p\to t>$
d. [[in]] = $\lambda y\lambda x[in(x, y)]$, $<in{:}r\to(r\to t)>$
e. in(G, S)

The treatment of a spatial relation given in (19d,e) fails to indicate which location stands for x and which for y. In fact, one of the difficulties with λ-operation is where to place its arguments. Example (19) can be treated more explicitly with Rule 1^{eq} equation solving.

(20) a. $\sigma(pl1)_t := \{x\}[x{=}G], <x{:}p\to t>$

b. $\sigma(pl2)_t := \{y\}[y{=}S], <y{:}p\to t>$

c. $\sigma(\mathbf{qslink})_t$
$:= \{x, y\}[\{[x{=}G]_t, [y{=}S]\} \oplus^{eq} in(x, y)]$
$:= \{x, y\}[[[x{=}G]_t \wedge [y{=}S]_t] \wedge in(x, y)]$

With the rule of substitution of identicals, we then obtain the same result $in(G, S)$, as given in (19e).

Rule 2 Distributive composition with the operator $\oslash$ applies to subordination or quantification constructions. Consider example (21), called *equi-NP construction*.[8]

(21) a. John$_{x1,w1}$ wants$_{e1,w2}$ to teach$_{e2,w4}$ on Monday.

b. Annotation (id = a$_{21}$):
Entity structures:
entity(x1, w1, *form*:John)
event(e1, w2, *pred*:want, **theme**(e1,e2))
event(e2, w4, *pred*:teach, **agent**(e2,x1))
Subordination link structure:
slink(e1, e2, modal)[9]

Pustejovsky et al. (2005) annotated the subordination relation between two events, *want(e_1)* and *teach(e_2)* as being *modal*. Montague Semantics, in contrast, treats it as a relation between the intensional predicate *want* and the property of teaching. However, the intensionality of the predicate *want* in the main clause requires Rule 2^i with an operator $\oslash^i$, a subtype of disjunctive composition for *intensional* cases like $\sigma(a_{21})$.

(22) a. Semantic forms of the entity structures:
$\sigma(x1)_t := \{x_1\}[x_1{=}John]$
$\sigma(e1)_{\mathcal{E}}$,where $\mathcal{E}{=}(v \to t)$,
$:= \{e_1, e_2\}[want(e_1) \wedge \mathbf{theme}(e_1, e_2)]$
$\sigma(e2)_{e\to(\mathcal{E}\to t)}$
$:= \{x_1, e_2\}[teach(e_2) \wedge \mathbf{agent}(e_2, x_1)]$

b. Semantic form of the subordination link structure:
$\sigma(slink)_t$
$:= \{x_1, e_1, e_2\}[[\{\sigma(e_1)_{\mathcal{E}}, \sigma(e_2)_{e\to(\mathcal{E}\to t)}]\oslash^i$
$(\sigma(e_1), \sigma(e_2))_{(e\to(\mathcal{E}\to t))\to(\mathcal{E}\to t)}]$
$:= \{x_1, e_1, e_2\}[\sigma(e_1)_t \to^{int} \sigma(e_2)_t]$
$:= \{x_1, e_1, e_2\}[[want(e_1) \wedge \mathbf{theme}(e_1, e_2)]$
$\to^i ([go(e_2) \wedge \mathbf{agent}(e_2, x_1)])]$

c. Semantic form of the whole annotation structure:
$\sigma(a_{21}) := \sigma(slink)_t$

The semantic form $\sigma(a_{21})$ shows that the predicate *want* has the event e_2 as its **theme** and that the **agent** of the predicate *go* in the subordinated complement is John. The non-Boolean connective $\to^{int}$ connects the semantic forms of the two components of the subordination construction (21) involving the *intensional* predicate *want*. The connective $\to^i$ needs to be defined as part of a model structure with a tentative definition as in (23):

(23) Definition of $\to^{int}$ (tentative)
Given a model M for a modal logic with a set W of possible worlds W that includes the actual world w_0 and an *intentional* world w_i accessible from w_0, and two semantic forms, ϕ and ψ, of type t,
$[\![\phi \to^i \psi]\!]^{M,w_0}$=1 iff
$[\![\psi]\!]^{M,w_i}$=1 provided $[\![\phi]\!]^{M,w_0}$=1.

This means that the eventuality of "teaching (on Monday)" is or becomes realized in the mind (intended world) of the experiencer *John* only.

[8]Annotation $\mathbf{a}_{21}$ is simplified to focus on the subordination link (**slink**).

[9]This example is taken from Pustejovsky et al. (2005), p. 553.

5. Model-theoretic Interpretation

5.1. General

Semantic forms are subject to a model-theoretic interpretation. Each well-formed semantic form $\sigma(\mathbf{a})$ of an annotation structure **a** is interpreted with respect to a model M and an assignment g of values to variables. $[\![\sigma(\mathbf{a})]\!]^{M,g}$ is then understood as the interpretation or denotation of $\sigma(\mathbf{a})$. The structure of each model M depends on the kind of semantic annotation. For the interpretation of temporal annotation, for instance, a set of times T and a set of temporal relations such as the precedence relation $\prec$ over T become a part of its model structure. Furthermore, the construction of such a model is constrained by some possible uses or definitions of logical predicates, called *meaning postulates*, as is discussed in 5.2.1.

5.2. Interpretation of unbound occurrences of variables

There may be some unbound occurrences of variables in well-formed semantic forms of *ABSr*. By Rule A.1 for Variable Registry, these variables may be either bound to the discourse referents registered before the semantic form of each of the substructures of an annotation structure or bound existentially when their scope is explicitly specified. Or else they can be interpreted with the assignment g as if they were bound existentially.

5.2.1. Meaning Postulates as Constraints

ABS makes use of logical predicates as part of the (object) representation language to simplify the representation of semantic forms or make it flexible to accommodate different interpretations. These predicates, marked in boldface, in *ABSr* are defined possibly in terms of higher-order logic as part of the model structure.
The predicate **past** is, for instance, introduced to represent the tense of an event as in (24):

(24) a. $[walk(e) \wedge \mathbf{past}(e)]$

b. instead of $[walk(e) \wedge e \subseteq t \wedge t \prec n]$

as in Kamp and Reyle (1993, page 521). Then its definition is given in (25) as part of an interpretation model structure.

(25) Truth Definition of Predicate **past**:
Given an event e, a runtime function τ from events to times, a time t, and the present time n, as specified in a model structure M,
past(e) is true with respect to a model M if and only if $\tau(e) \subseteq t$ and $t \prec n$.

The predicate **past** may be defined differently to accommodate its deitic or situational use (see Partee (1973) or Quirk et al. (1985)).
Aspectual features such as *present perfect* and *progressive* are also encoded into annotations just as they are. Consider a case of the present perfect aspect in (26).

(26) a. Mia [has visited]$_{e1}$ Boston.

b. Annotation (id=a_{26}):
event (e1, w2-3, *pred*:visit, *tense*:present,
aspect: perfect)

c. Semantic Form:
$\sigma(e1) := [visit(e_1) \wedge \mathbf{presPerfect}(e_1)]$

Semantic form (26c) is then interpreted by the definition of **presPerfect** given as part of a model structure. Otherwise, its representation gets complicated similar to DRS, for instance. Here is an example from Cann et al. (2009).

(27) a. The plant has died.

b. {a, e, t, n, r, s, u}
$e \subseteq t$
$t \leq n$
r=n
Result-from'(e, s)
$s \bigcirc r$
Die'(e, u)
u=a
Plant'(u)
Dead'(s, u)

ABSr, in contrast, yields the following representation:

(28) a. The plant has died.

b. Annotation:
entity(x1, w2, *type*:plant)
event(e1, w4, *pred*:die, *tense*:present,
aspect:perfct)
srlink(e1,x1, theme)

c. Semantic Forms:
$\sigma(x_1) := plant(x_1)$
$\sigma(e_1) := [die(e_1) \wedge \mathbf{presPerfect}(e_1)]$
$\sigma(srlink)$
$:= [\{\sigma(x_1)_t, \sigma(e_1)_t\} \oplus^{bo} \mathbf{theme}(e_1, x_1)_t]$
$\sigma(_{26})$
$:= \{e, x\}[die(e) \wedge \mathbf{presPerfect}(e) \wedge$
$\mathbf{theme}(e, x)]$

The interpretation of $\sigma(e_1)$ in (28c), for instance, requires the truth-conditional definition of **presPerfect**(e) that reflects those notions of the perfective aspect encoded in DRS (27b) above.
Furthermore, the proposed way of treating tense, aspect, and other complex predicates allows different interpretations or uses of them. Those predicates that constitute part of the representation language of semantic forms in *ABSr*, however, require truth-definitions or *meaning postulates* that constrain and define a set of admissible model structures (see Carnap (1947 1956; Montague (1974; Dowty (1979)).

6. Applications

6.1. Boolean Conjunctive Composition

ISO-Space (ISO, 2020) introduces the movement link (**movelink**) to annotate motions involving paths. The predicate **traverses** associated with motions is one of the logical predicates that need to be defined in the model structure of *ABS*. It can also be illustrated how the semantic forms involving motions and paths can be derived through Rule 1^{bo} Boolean conjunctive composition, as is demonstrated in (29).

(29) a. Marakbles:
Mia$_{x1,w1}$ arrived$_{m1,w2}$ $\emptyset_{ep1}$ in Boston$_{pl1,w4}$ yesterday.

b. Annotation (id=a$_{29}$):
Entity structures:
entity(x1,w1, *type*:person, *form*:nam)
motion(m1,w2, *pred*:arrive, *type*: transition,
tense:past)
eventPath(ep1,$\emptyset$, *start*:unspecified, *end*:pl1,
trigger(m1,ep1))
place(pl1,w4, *type*:city, *form*:nam)
Movement link structure:
movelink(*figure*:x1, *ground*:ep1,
relType:traverses)

Each markable is identified with an ID associated with its category and anchored to a word. Motions, as denoted by verbs like `arrive`, trigger a path, called *event-path*. This path is marked with a null category or non-consuming tag $\emptyset$ because it is not associated with any non-null string of words.

(30) a. Semantic forms of entity structures:
$\sigma(x1)_t := [person(x_1) \wedge named(x_1, Mia)]$
$\sigma(m1)_t := [arrive(m_1) \wedge past(m_1)]$
$\sigma(ep1)_t := [start(\pi, \gamma(l_0)) \wedge end(\pi, l_1)$
$\wedge \mathbf{triggers}(m_1, \pi)]$
$\sigma(pl1)_t := [named(l_1, Boston) \wedge city(l_1)]$

b. Semantic form of the movement link structure:
$\sigma(movelink)$
$:= [\{\sigma(x1)_t, \sigma(ep1)_t\} \oplus^{bo} \mathbf{traverses}(x, \pi)_t]$
$:= [[[person(x_1) \wedge named(x_1, Mia)]$
$\wedge [start(\pi, \gamma(l_0)) \wedge end(\pi, l_1)$
$\wedge \mathbf{triggers}(m_1, \pi)]$
$\wedge [named(l_1, Boston) \wedge city(l_1)]]$
$\wedge \mathbf{traverses}(x, \pi)]$

c. Annotation structure:
$\sigma(a_{29})$
$:= \{x_1, \pi_1, l_0, l_1, m_1\}\sigma(movelink)$
$=: \{x, \pi, l_0, l_1, m\}$
$[[[person(x) \wedge named(x, Mia)]$
$\wedge [start(\pi, \gamma(l_0)) \wedge end(\pi, l_1)$
$\wedge \mathbf{triggers}(m, \pi)]$
$\wedge [named(l_1, Boston) \wedge city(l_1)]]$
$\wedge \mathbf{traverses}(x, \pi)]$

All of the semantic forms that are derived through various links have been shown to undergo Rule 1^{bo} Boolean conjunctive composition only. This was illustrated with **srlink** for semantic role labeling, **tlink** for temporal anchoring, **qslink** for the location of regions, and **movelink** for the annotation of motions involving their movers and event-paths.

6.2. Distributive Composition for Conditionals

Besides its subtype $\oslash^{int}$ for intensional subordinate constructions, the distributive composition can have other subtypes. Here I introduce Rule 2^{imp} with the operator $\oslash^{imp}$ for the case of implication. The word *if* in English triggers a conditional sentence which is often interpreted as a

truth-functional implication in Propositional Logic. Given two well-formed formulas ϕ and ψ, the conditional formula $[\phi \rightarrow \psi]$ is treated as a well-formed formula in Propositional Logic and interpreted truth-functionally as being false only if ϕ is true but ψ is false. Although the interpretation of conditionals in ordinary language is more complex than the truth-functional interpretation just given, (31) and (32) illustrate how *if*-constructions are annotated and how their semantic forms are represented in a tripartite structure.

(31) Data:
If it rains tomorrow, then the picnic will be canceled.

(32) a. Annotation of Antecedent (id=a_{32a}):
event(e1, w3, *pred*: rain)
timex3(t1, w4, *type*:date, *value*:2020-02-04)
tlink(tl1, e1, t1, isIncluded)

b. Annotation of Consequent (id=a_{32b}):
event(e2, w7, *pred*: picnic)
event(e3, w10, *pred*: beCanceled, *tense*:future, *theme*:e2)
timex3(t2, $\emptyset$, *type*:date, *value*:unspecified)
tlink(tl2, e3, t2, isIncluded)

c. Subordination link:
slink(*antecedent*:a1, *consequent*:a2, conditional)

Based on annotation (32), we obtain the semantic forms,as shown in (33):

(33) a. Semantic forms of antecedent:
$\sigma(e1)_t := [rain(e_1)_t]$
$\sigma(t1)_t := [date(t_1) =$2019-02-04$]_t$
$\sigma(tl1)$
$:= [\{\sigma(e1)_t, \sigma(t1)_t\} \oplus^{(} bo) occurs(e_1, t_1)_t]$
$:= [[rain(e_1) \wedge date(t_1,$2019-02-04$)]$
$\wedge occurs(e_1, t_1)]_t$

b. Semantic form of consequent:
$\sigma(e2)_t := [picnic(e_2)]$
$\sigma(e3)_t := [beCanceled(e_3) \wedge theme(e_3, e_2)]$
$\sigma(t2)_t := \gamma(t_2)$[10]
$\sigma(tl2)$
$:= [\{\sigma(e3)_t, \sigma(t_2)_t\} \oplus^{bo} occurs(e_3, \gamma(t_2))_t]$
$:= [[[beCanceled(e_3) \wedge theme(e_3, e_2)]$
$\wedge \gamma(t_2)] \wedge occurs(e_3, \gamma(t_2))]_t$

c. Semantic form of conditional:
$\sigma(slink)$
$:= [\{\sigma(tl_1)_t, \sigma(tl_2)_t\} \oslash^{imp}$
$implies(\sigma(tl_1), \sigma(tl_2))_{t\rightarrow(t\rightarrow t)}]$
$:= [\sigma(tl_1) \rightarrow \sigma(tl_2)]$
$:= [[rain(e_1) \wedge date(t_1,$2019-02-04$)$
$\wedge \mathbf{occurs}(e_1, t_1)]_t \rightarrow [[beCanceled(e_3)$
$\wedge theme(e_3, e_2) \wedge \mathbf{future}(e_3)] \wedge \gamma(t_2)$
$\wedge \mathbf{occurs}(e_3, \gamma(t_2))_t]]$

d. $\sigma(a_{32b})$
$:= \{e_1, e_2, e_3.t_1, \gamma(t2)\}\sigma(slink)$
$[[rain(e_1) \wedge date(t_1,$2019-02-04$)$
$\wedge \mathbf{occurs}(e_1, t_1)]_t \rightarrow [[beCanceled(e_3)$
$\wedge theme(e_3, e_2) \wedge \mathbf{future}(e_3)] \wedge \gamma(t_2)$
$\wedge \mathbf{occurs}(e_3, \gamma(t_2))_t]]$

With respect to the operator $\oslash^{imp}$, the semantic form of the antecedent, $\sigma(tl1)$, is understood to be the restrictor R and that of the consequent, $\sigma(tl2)$, is the nuclear scope N, while the relation of implication between them is represented by the operator $\rightarrow$.

7. Comparison

7.1. Related Work

There have been several theoretical works showing how annotation structures can be interpreted and a variety of large-scale computational efforts to implement them for computational applications. Some of them are annotation-based semantics in one way or another.

Hobbs and Pustejovsky (2003) develop a semantics for TimeML (Pustejovsky et al., 2005), based on the OWL-time ontology. They provide a fine-grained way of annotating and interpreting various temporal relations. *ABS* is designed to accommodate the OWL-time ontology in defining its logical predicates related to temporal annotation.

Katz (2007) introduces a denotational semantics that directly interprets TimeML annotation structures represented in XML. The model structure proposed in Katz (2007) becomes part of the temporal model structure for *ABS* .

Bunt (2007) and Bunt (2011) introduce a semantics for semantic annotation. This eventually develops into a semantics based on the abstract syntax of a semantic annotation scheme. Bunt (2020a) and Bunt (2020b) have developed QuantML, a markup language for quantification, that can apply to the annotation and interpretation of a full-range of features related to quantification such as the definiteness, involvement or collectivity (distributivity) of entities or scope ambiguity involving quantifiers and eventualities.

Lee (2008) and Lee (2011) follow the OWL-time ontology and a compositional approach to work on temporal annotations with an extensive use of λ-operations. It shows some degree of complexity in the use of λ-operations when they are recursively embedded, for it requires to raise the order of variables as the embedding gets deeper.

One of the reasons for introducing *ABSr* is to avoid recursive embedding and substitutions (see Hausser (2015)). For now, *ABSr* has Rule 1^{sub} Substitutive conjunctive composition, but this should be deleted eventually except for the illustration of rudimentary annotations involving names and other basic types. Database Semantics (DBS) (Hausser, 2006) provides a theoretical foundation for the understanding of language analysis and generation without recursions and substitutions, but with the associative linear processing of language. This has motivated the design of *ABS* to some extent.

Then there are other types of semantics that present different ways of representing meaning in language. Banarescu et al. (2013) introduce AMR (the Abstract Meaning Representation) to represent the semantic roles mainly based on

[10] γ is a function that assigns a time to a deitic temporal expression or a contextually determinable unspecified time.

PropBank in a logical format, PENNMAN format, or directed graph structure. He (2018) also introduces a way of annotating semantic roles, which is called *Shallow Semantics*, without relying on pre-defined syntactic structures but introducing syntax-independent span-based neural models or labelled span-graph networks (LSGNs).

Based on syntax-free annotations, *ABSr* is also syntax-independent. Its current representation format is strictly linear but needs to move onto a graphic mode for visual purposes. The composition rules of *ABSr* are constrained by type matching and also syntax-independent unlike Moens and Steedman (1988)'s categorial grammar or Kamp and Reyle (1993)'s DRSs. Dobnik et al. (2012) and Dobnik and Cooper (2017) introduce a type theory with records to constrain semantic representations and their manipulations in language processing. Their type system, especially related to spatial perception, will properly orient the spatio-temporal annotation of ISO-Space and meaning representation through *ABS*. The earlier work of Pustejovsky (2001) on type construction also lays a basis for the type theory of *ABS* for a finer-grained treatment of entities and eventualities.

For the computational applications of semantic annotations, the Gronigen Meaning Bank (GMB) (Bos et al., 2017) is very much related to the basic motivation of *ABS* in efforts to modify the classical version of DRT by making its syntax based on a (Montagovian) type systems consisting of two types, e and t, and by translating DRSs into a first-order logic only, for instance, while deleting so-called *duplex conditions* in DRSs. The basic design of the Parellel Meaning Bank (PMB) also adopts DRT as its formalism for meaning representation while adopting Combinatory Categorial Grammar as its syntax. Since it applies to multilingual annotation, *ABS* can make use of it when the ISO standards on semantic annotation are extended to multilingual annotations, especially for the purposes of multilingual translations.

Nevertheless, the theoretical framework of *ABS* and its representation language is conservative in practice, being essentially based on the λ-calculus and the graphic representation of Kamp and Reyle (1993)'s DRT. This will be shown in the ensuing Subsection 7.2.

7.2. Convertibility

The composition of semantic forms is constrained by their semantic types. These types simply reflect those in Montague semantics (Montague, 1974) and (Dowty et al., 1981) and also the extended type theory by Kracht (2002) and Pustejovsky et al. (2019), thus making all these semantic forms isomorphic to those λ-constructions in λ-calculus. If such a typing of the semantic forms of annotation structures is ignored or if each of the semantic forms is treated as being of type t, then these semantic forms can easily be converted to DRSs (Kamp and Reyle, 1993).

There is an option to choose a type-theoretic semantics or not. *ABS* allows both but prefers to choose a type-theoretic semantics to constrain its representation language *ABSr*, while enriching its interpretation model structure, as shown in Figure 2.[11]

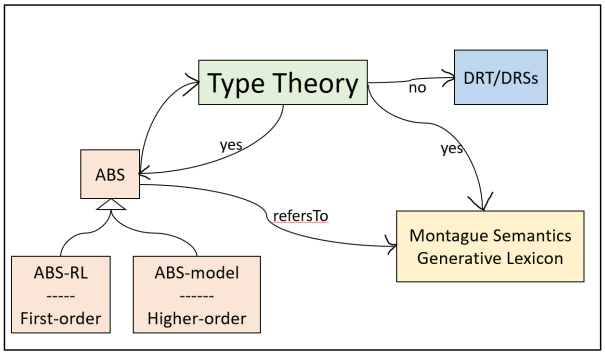

Figure 2: Options: Type-theoretic or Not

If a type theory is adopted, then the logical predicates can be defined in terms of type-theoretic higher-order logic.

In *ABS*, the choice of a theory depends on the treatment of unbound variables and unspecified types. *ABS* treats logical forms with occurrences of unbound variables as well-formed semantic forms. Individual (or predicate) variables may occur unbound in well-formed semantic forms, as in the interval temporal logic of Pratt-Hartmann (2007).[12] Here is an example with a markable "visited"$_{e_1}$:

(34) a. Data:
Mia$_{x1}$ visited$_{e_1}$ Berlin, New York, [last year]$_{t1}$.

b. Annotation (id=a$_{5.unbound}$):
Entity structures:
event(e1, m1, *pred*:visit, *tense*:past)
timex3(t1, m2, *type*:gYear, *value*:2019)
Link structure:
tlink(e1, t1, isIncluded)

c. Semantic Forms:
$\sigma(e1)_\alpha := \{e_1\}[visit(e_1) \wedge past(e_1)]$
$\sigma(t1)_\beta := \{t_1\}[gYear{=}(t_1, 2019)]$
$\sigma(tlink)_\gamma$
$:= \{e_1, t_1\}[\{\sigma(e1), \sigma(t1)\} \bigcirc \mathbf{occurs}(e_1, t_1)]$

Each of the semantic forms in (34c) contains some variables which are registered in its preamble. In *ABSr*, these variables can be bound in two different ways, either by the existential quantifier or by the λ-operator. The assignment of a type to each semantic form depends on which way these (registered) variables are bound. The type of each semantic form is:

- Case 1: either of type t (truth-value carrying) as if the unbound variables were bound by the existential quantifier $\exists$:
i.e., $\exists\{e\}[visit(e) \wedge past(e)]$ (type t)
- Case 2: or of some functional type (predicate) as if the unbound variables were bound by the λ-operator:
i.e., $\lambda e[visit(e) \wedge past(e)]$ (type $v \rightarrow t$)

[11] Although Figure 2 indicates that DRT/DRSs are not based on a type theory, the DRT formalism adopted by Bos et al. (2017) is based on a type theory.

[12] *ABS* has no predicate variables.

Depending on which case is chosen, the semantic form of a link like $\sigma(tlink)$ in (34c) undergoes a different rule of composition.

Case 1 allows the conversion of semantic forms in *ABS* to DRSs.

(35) Case 1:
Rule 1 Boolean conjunctive composition
a. $\sigma(tlink)$
$:= [\{\sigma(e_1)_t, \sigma(t_1)_t\} \oplus^{bo} occurs(e_1, t_1)_t]$
$:= \{e,t\}[[visit(e) \wedge past(e)] \wedge gYear(t, 2019)$
$\wedge occurs(e,t)]$
b. $\sigma(a_{34})$ = $\sigma(tlink)$

As shown in (35), Case 1 Boolean conjunctive composition ($\oplus^{bo}$) can easily be converted to an equivalent DRS.

(36) Case 1 in DRS:

e t
visit(e)
past(e)
gYear(t,2019)
occurs(e,t)

Although the application of Rule 1^{bo} Boolean conjunctive composition is type-constrained, there is no such a constraint on the derivation of DRSs.

Case 2 allows the conversion of semantic forms in *ABSr* to well-formed forms in λ-calculus as in Montague Semantics (Montague, 1974). For the illustration of Case 2, consider example (34), as was just given:

(37) Case 2:
Rule 2 Functional conjunctive composition ($\oplus^{fa}$):
a. $\sigma(tlink)_t$
$:= [\{\sigma(e_1)_\mathcal{E}, \sigma(t_1)_I\} \oplus^{fa} occurs(e_1, t_1)_{I\rightarrow(\mathcal{E}\rightarrow t)}]$
$:= [[visit(e_1) \wedge past(e_1)] \wedge gYear(t_1,2019)$
$\wedge occurs(e_1, t_1)]$
b. $\sigma(a_{34})$ = $\sigma(tlink)_t$

The semantic form $\sigma(tlink)$ in (37) is treated of a functional type, $I \rightarrow (\mathcal{E} \rightarrow t)$, where I is $i \rightarrow t$ and $\mathcal{E}$ is $v \rightarrow t$. Then the semantic forms $\sigma(e1)$ and $\sigma(t1)$ are treated as arguments of $\sigma(tlink)$ such that they are of types $\mathcal{E}$ (set of eventuality descriptions) and I (set of time points), respectively.
In the process of the Boolean conjunctive composition, the unbound occurrences of the variables are anchored to the discourse referents e and t, as in DRS, or existentially quantified, while adjusting their variable names accordingly.
As for the case of the functional conjunctive composition, the whole process is understood as if all the semantic forms were subject to a series of λ-conversions as in (38):

(38) λ-operations:
a. $\sigma(e_1)_{v\rightarrow t} := \lambda e_1[visit(e_1) \wedge past(e_1)]$
b. $\sigma(t_1)_{i\rightarrow t} := \lambda t_1[gYear(t_1,2019)]$
c. $\sigma(tlink)_t :=$
$\lambda T \lambda \mathcal{E} \exists \{e,t\}[\mathcal{E}(e) \wedge T(t) \wedge occurs(e,t)]$
$(\sigma(e_1))(\sigma(t_1))$
$:= \exists\{e,t\}[\sigma(e1)(e) \wedge \sigma(t1)(t)]$
$:= \exists\{e,t\}[[visit(e) \wedge past(e)] \wedge gYear(t,2019) \wedge$
$occurs(e,t)]$

It should again be stated that the derivation of semantic forms in *ABSr* does not undergo such λ-operations. The application of Rule 2 Functional conjunctive composition is only implicitly understood to undergo such operations.
Unlike semantic forms that involve λ-operations, the application of the $\oplus^{fa}$ in *ABSr* does not introduce predicate variables of a higher-order, but individual variables of the first order only. This keeps *ABSr* to remain at the level of first-order.

8. Concluding Remarks

As in other parts of ISO 24617 standards on semantic annotation, this paper has a gap in dealing with the semantics of entities and determiners that include generalized quantifiers. Specifically, this paper fails to fully accommodate the new developments on quantification that have been made by Bunt (2020a) and Bunt (2020b).
ABS aims to lighten the burden and possible complexity of generating semantic annotation structures. It would be an ideal situation if semantic annotation structures could have every piece of relevant semantic information encoded into them and be interpreted directly without relying on any intermediate auxiliary representation scheme. But the task of generating such annotation structures and interpreting them directly should easily run into enormous cost and complexity.
ABS is an annotation-based semantics that converts annotation structures to semantic forms for their (model-theoretic) interpretation. For the representation of these semantic forms, *ABS* provides a simple representation language, a type-theoretic first-order logic without the overuse of λ-operations. This language makes use of a small set of *logical* predicates, such as referring to semantic roles or event and time structures and types, that are defined as part of an interpretation model. The meta-language that defines these logical predicates may be of a higher-order logic.
To follow the principle of semantic compositionality, *ABS* introduces two types of composition with the conjunctive $\oplus$ and distributive $\oslash$ operators and their subtypes over the semantic forms of annotation structures that consist of entity and link structures. Most, if not all, of the link structures in ISO-TimeML and ISO-Space only require conjunctive composition, while quantificational, plural constructions or some subordinated constructions such as the *if-then* construction may undergo distributive (selective) composition.
There are two major types of conjunctive composition: the Boolean type $\oplus^{boo}$ and the functional type $\oplus^{fa}$. Then the functional type has two subtypes, one by substitution $\oplus^{sub}$ and the other by equation solving $\oplus^{eq}$. Annotation structures that are isomorphic to non-embedded structures in Kamp and Reyle (1993)'s DRSs are considered as undergoing the process of Boolean conjunctive composition. In contrast, those annotation structures that match λ-structures in Montague Semantics (Montague, 1974) undergo the functional conjunctive composition. This distinction is not very significant, for the semantic forms of most

of the annotation structures undergo the process of Boolean conjunctive composition only.

This is the first version of *ABS*. It requires to be further tested against a variety of larger data and annotation structures. This should be the case especially for the distributive composition involving complex semantic structures.

9. Acknowledgements

Thanks to Jae-Woong Choe, Chongwon Park, and James Pustejovsky for their reading the preliminary draft with invaluable comments and to the four anonymous reviewers for their detailed constructive comments. I am very much indebted to Harry Bunt for his laborious work to help improve the final submission for publication. I thank them all, but do not claim that all these reviewers agree with my proposal or that I have fully succeeded in accommodating their comments and suggestions.

10. Bibliographical References

Allen, J. F. (1984). Towards a general theory of action and time. *Artifical Intelligence*, 23:123–54.

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., and Schenider, N. (2013). Abstract meaning representationf or sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria, August.

Bos, J., Basile, V., Evang, K., Venhuizen, N. J., and Bjerva, J. (2017). The Groningen Meaning Bank. In Nancy Ide et al., editors, *Handbook of Linguistic Annotation*, pages 463–496. Springer, Berlin.

Bunt, H. (2007). The semantics of semantic annotations. In *Proceedings of the 21st Pacific Asia Conference on Language, Information and Computation*, pages 13–28, Seoul, Korea. The Korea Society for Language and Information.

Bunt, H. (2010). A methodology for designing semantic annotation languages exploiting semantic-syntactic isomorphisms. In Alex C. Fang, et al., editors, *Proceedings of the Second International Conference on Global Interoperability for Language Resources (ICGL20100)*, pages 29–46, City University of Hong Kong, Hong Kong.

Bunt, H. (2011). Introducing abstract syntax + semantics in semantic annotation, and its consequences for the annotation of time and events. In Eunryoung Lee et al., editors, *Recent Trends in Language an Knowledge Processing*, pages 157–204. Hankookmunhwasa, Seoul.

Bunt, H. (2020a). Annotation of quantification: the current state of ISO 24617–12. In Harry Bunt, editor, *Proceedings of the 16th Joint ISO–ACL/SIGSEM Workshop on Interoperable Semantic Annotation*, pages 1–13, May. A satellite workshop at LREC 2020, May 11–15, 2020, Marseille, France (postponed due to COVID–19).

Bunt, H., (2020b). *Semantic Annotation of Quantification in Natural Language*. TiCC/Department of Cognitive Science and Artificial Intelligence, Tilburg University, Tilburg, 2nd edition, February. TiCC TR 2020-2.

Cann, R., Kempson, R., and Gregoromichelaki, E. (2009). *Semantics: An Introduction to Meaning in Language*. Cambridge University Press, Cambridge.

Carnap, R. (1947, 1956). *Meaning and Necessity: A Study in Semantics and Modal Logic*. The University of Chicago Press, Chicago, 2nd edition.

Copestake, A., Flickinger, D., Sag, I., and Pollard, C. (2005). Minimal recursion semantics: an introduction. *Journal of Research on Language and Computation*, pages 281–332.

Davidson, D. (1979). The logical form of action sentences. In N. Rescher, editor, *The Logic of Decision and Action*, pages 81–120, Pittsburgh. University of Pittsburgh Press. Reprinted in Davidson (2001).

Davidson, D. (2001). *Essays on Actions and Events*. Oxford University Press, Oxford, 2nd edition.

Dobnik, S. and Cooper, R. (2017). Interfacing language, spatial perception and cognition in type theory with records. *Journal of Language Modelling*, 5(2):273–301.

Dobnik, S., Cooper, R., and Larsson, S. (2012). Modelling language, action, and perception in type theory with records. In D. Duchier et al., editors, *Constraint Solving and Language Processing - 7th International Workshop on Constraint Solving and Language Processing, CSLP 2012*, Orelans, France, September. Revised Selected Papers, number 8114 in Publications on Logic, Language and Information (FoLLI), Springer, Berlin, Heidelberg, 2013.

Dowty, D. R., Wall, R. E., and Peters, S. (1981). *Introduction to Montague Semantics*. D. Reidel, Dordrecht.

Dowty, D. R. (1979). *Word Meaning and Montague Grammar: The Semantics of Verbs and Times in Generative Semantics and in Montague's PTQ*. D. Reidel, Dordrecht.

Gordon, A. S. and Hobbs, J. R. (2017). *A Formal Theory of Common Sense Psychology: How People Think People Think*. Cambridge University Press, Cambridge.

Hausser, R. (2006). *A Computational Model of Natural Language Communication: Interpretation, Inference, and Production in Database Semantics*. Springer, Berlin.

Hausser, R. (2015). From montague grammar to database semantics. *Language and Information*, 19(2):1–16. available at *lagrammar.net*.

He, L. (2018). *Annotating and Modeling Shallow Semantics Directly from Text*. Dissertation of doctor of philosophy in computer science and engineering, University of Washington.

Hobbs, J. and Pustejovsky, J. (2003). Annotating and reasoning about time and events. In *Proceedings of AAAI Spring Symposium on Logical Formalizations of Common Sense Reasoning*, Stanford, CA, March. Reprinted in Mani et al. (eds), 2005, pages 301-315.

ISO, (2012). *ISO 24617-1 Language resource management - Semantic annotation framework - Part 1: Time and events*. International Organization for Standardization, Geneva. Working group: ISO/TC 37/SC 4/WG 2 semantic annotation.

ISO, (2014). *ISO 24617-4 Language resource management - Semantic annotation framework - Part 4: Semantic roles (SemAF-SR)*. International Organization for Stan-

dardization, Geneva. Working group: ISO/TC 37/SC 4/WG 2 semantic annotation.

ISO, (2020). *ISO 24617-7 Language resource management - Semantic annotation framework - Part 7: Spatial information*. International Organization for Standardization, Geneva, 2nd edition. Working group: ISO/TC 37/SC 4/WG 2 semantic annotation.

Kamp, H. and Reyle, U. (1993). *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer Academic Publishers, Dordrecht.

Katz, G. (2007). Towards a denotational semantics for TimeML. In Frank Schilder, et al., editors, *Annotating, Extracting and Reasoning about Time and Events*, pages 88–106, Berlin. Springer.

Kracht, M. (2002). On the semantics of locatives. *Linguistics and Philosophy*, 25:157–232.

Lee, K., Pustejovsky, J., and Bunt, H. (2018). Revising ISO-Space and the role of the movement link. In Harry Bunt, editor, *Proceedings of the 14th Joint ACL–ISO Workshop on Interoperable Semantic Annotation (ISA-14): COLING 2018 Workshop*, pages 35–44, Santa Fe, New Mexico, U.S.A, August.

Lee, K. (1983). Equation solving. In Chungmin Lee et al., editors, *Language, Information and Computation*, pages 14–26. Taehaksa, Seoul.

Lee, K. (2008). Formal semantics for interpreting temporal annotation. In Piet van Sterkenburg, editor, *Unity and Diversity of Languages*, pages 97–108, Amsterdam. John Benjamins Publishing Co. Invited talk at the 18th Congress of Linguists, held in Seoul on July 21–26 2008.

Lee, K. (2011). A compositional interval semantics for temporal annotation. In Eunryoung Lee et al., editors, *Recent Trends in Language an Knowledge Processing*, pages 122–156. Hankookmunhwasa, Seoul.

Lee, K. (2016). An abstract syntax for ISO-Space with its `<moveLink>` reformulated. In Harry Bunt, editor, *Proceedings of the LREC 2016 Workshop ISA-12 – 12th Joint ACL-ISO Workshop on Interoperable Semantic Annotation*, pages 28–37, Portorož, Slovenia, May.

Lee, K. (2018). Revising ISO-Space for the semantic annotation of dynamic spatial information in language. *Language and Information*, 22.1:221–245.

Link, G. (1998). *Algebraic Semantics in Language and Philosophy*. CSLI Publications, Stanford, CA.

Mani, I. and Pustejovsky, J. (2012). *Interpreting Motion: Grounded Representations for Spatial Language*. Oxford University Press, Oxford.

Miller, R. and Shanahan, M. (1999). The event-calculus in classical logic — alternative axiomatizations. *Electronic Transactions on Artificial Intelligence*, 3(1):77–105.

Moens, M. and Steedman, M. (1988). Temporal ontology and temporal reference. *Computational Linguistics*, 14(2):15–28.

Montague, R. (1974). *Formal Philosophy: Selected Papers of Richard Montague*. Yale University Press, New Haven and London.

Parsons, T. (1990). *Events in the Semantics of English: A Study in Subatomic Semantics*. The MIT Press, Cambridge, MA.

Partee, B. H. (1973). Some structural analogies between tenses and pronouns in English. *The Journal of Philosophy*, 80(18):601–9. Reprinted in *Compositionality in Formal Semantics: Selected Papers by Barbara H. Partee*, Malden, MA: Blackwell. pp. 50–58.

Pratt-Hartmann, I. (2007). From TimeML to interval temporal logic. In Harry Bunt, editor, *Proceedings of the Seventh International Workshop on Computational Semantics*, pages 111–180, Tilburg, the Netherlands. Tilburg University.

Pustejovsky, J., Ingria, R., Saurí, R., o, J. C., Littman, J., Gaizauskas, R., Setzer, A., Katz, G., and Mani, I. (2005). The specification language TimeML. In James Pustejovsky Inderjeet Mani et al., editors, *The Language of Time*, pages 545–557. Oxford University Press, Oxford.

Pustejovsky, J., Lee, K., Bunt, H., and Romary, L. (2010). ISO-TimeML: An international standard for semantic annotation. In Harry Bunt, editor, *Proceedings of LREC 2010*, Valletta, Malta, May. LREC 2010.

Pustejovsky, J., Lee, K., and Bunt, H. (2019). The semantics of ISO-Space. In Harry Bunt, editor, *Proceedings of the 15th Joint ACL – ISO Workshop on Interoperable Semantic Annotation (ISA-15)*, pages 46–53, Gothenburg, Sweden, May. International Workshop on Computational Semantics (IWCS 2029).

Pustejovsky, J. (1995). *The Generative Lexicon*. The MIT Press, Cambridge, MA.

Pustejovsky, J. (2001). Type construction and the logic of concepts. In Pierrette Bouillon et al., editors, *The Language of Word Meaning*, pages 91–135. Cambridge University Press, Cambridge, UK.

Quirk, R., Greenbaum, S., Leech, G., and Svartvik, J. (1985). *A Comprehensive Grammar of the English Language*. Longman, London and New York, January.