Comparing DBS and Predicate Calculus

Roland Hausser

Universität Erlangen-Nürnberg (em.)

Abstract

A content like Fido found a bone. may be viewed from two basic perspectives. One takes the view of an outside observer. For example, Predicate Calculus analyzes the truth relation between a formal sign and a formal model (1) or model structure (8). This approach is called *sign-based*.

The other takes the view of the agent, i.e., Fido looking, listening, and sniffing out into the world, keeping track of it, and deriving suitable action. For example, DBS models the cognition of an artificial agent with an on-board interface component for automatically monitoring recognition (11) and action (12), and a content-addressable, on-board memory component for storing and retrieving content. Because the agent's processing of sensory input and output is central to this approach, it is called *agent-based*.[1]

keywords: over-generation, under-generation, coreference by address, donkey sentence, model, model structure, propositions being content vs. denoting truth values

1 Definition of Predicate Calculus

Montague's (1973) version of first order Predicate Calculus, hence abbreviated PredC, is widely admired.[2] Leaving the intension-extension distinction and the use of lambda calculus aside, it may be presented as follows (slightly revised):

1.1 FORMAL DEFINITION OF PREDC

1. A model $\mathscr{M}$ is defined as the quadruple <A, B, F, g>, where A is an infinite set of objects or individuals, B a finite set of basic expressions, F a denotation function from B into the free monoid[3] A^* over A, and g an assignment function from variables into A^*.
2. The elements $\emptyset$ (empty set) and $\{\emptyset\}$ (set containing the empty set) of A^* are used as the denotation of the truth values 1 (true) and 0 (false), respectively.
3. Syntactically, the operators $\neg$, $\wedge$, $\vee$, =, $\rightarrow$, $\forall$, and $\exists$ are defined as follows:

   (a) If f is a one-place functor and α is a name, then $f(\alpha)$ is a sentence.
   (b) If ϕ is a sentence, then $\neg\phi$ is a sentence.

[1] For earlier attempts at integrating symbolic logic into an agent-based cognition see Hausser (1980), Kamp (1980), and Barwise and Perry (1983).

[2] Thanks to Professors Nuel Belnap, Georg Kreisel, and Rich Thomason for deepening my understanding of formal semantics and Montague Grammar. Thanks also to Professor Moravcsik for two one-year stays at the Stanford Philosophy Department in 1979-80 and 1983-84.

[3] The * is called the Kleene Star. The free monoid A^* serves as the "universe of discourse."

(c) If ϕ is a sentence and ψ is a sentence, then $\phi \wedge \psi$ is a sentence.
(d) If ϕ is a sentence and ψ is a sentence, then $\phi \vee \psi$ is a sentence.
(e) If ϕ is a sentence and ψ is a sentence, the $\phi \rightarrow \psi$ is a sentence.
(f) If ϕ is a sentence and ψ is a sentence, then $\phi = \psi$ is a sentence.
(g) If f and h are functors and x is a variable, then $\exists\mathsf{x}[\mathsf{f}(\mathsf{x}) \wedge \mathsf{h}(\mathsf{x})]$ is a sentence.
(h) If f and h are functors and x is a variable, then $\forall\mathsf{x}[\mathsf{f}(\mathsf{x}) \rightarrow \mathsf{h}(\mathsf{x})]$ is a sentence.

Definitions (a–f) constitute Propositional Calculus, hence abbreviated PropC. Definitions (g–h) extend PropC into PredC by introducing the quantifiers $\exists\mathsf{x}$ and $\forall\mathsf{x}$ binding a variable. Within a formula, the *scope* of a quantifier is the area in which its variable is bound; the area is defined by the formula's bracketing structure.

4. Semantically, the set of operators is defined as follows:

(a) If f is a one-place functor, then $\mathsf{f}(\alpha)$ is a true sentence relative to a model $\mathscr{M}$ iff if the denotation of α in $\mathscr{M}$ is an element of the denotation of f in $\mathscr{M}$.

(b) $\neg\ \phi$ is a true sentence relative to a model $\mathscr{M}$ if and only if the denotation of ϕ is 0 relative to $\mathscr{M}$.

(c) $\phi \wedge \psi$ is a true sentence relative to a model $\mathscr{M}$ if and only if the denotations of ϕ and of ψ are 1 relative to $\mathscr{M}$.

(d) $\phi \vee \psi$ is a true sentence relative to a model $\mathscr{M}$ if and only if the denotation of ϕ or ψ is 1 relative to $\mathscr{M}$.

(e) $\phi \rightarrow \psi$ is a true sentence relative to a model $\mathscr{M}$ iff the denotation of ϕ is 0 relative to $\mathscr{M}$ or the denotation of ψ is 1 relative to $\mathscr{M}$.

(f) $\phi = \psi$ is a true sentence relative to a model $\mathscr{M}$ if and only if the denotation of ϕ relative to $\mathscr{M}$ equals the denotation of ψ relative to $\mathscr{M}$.

(g) $\exists\mathsf{x}[\mathsf{f}(\mathsf{x}) \wedge \mathsf{h}(\mathsf{x})]$ is a true sentence relative to $\mathscr{M}$ and a variable assignment g iff at least one $\mathsf{g}'(\mathsf{x})$ makes $[\mathsf{f}(\mathsf{x}) \wedge \mathsf{h}(\mathsf{x})]^{\mathscr{M},g}$ true.

(h) $\forall\mathsf{x}[\mathsf{f}(\mathsf{x}) \rightarrow \mathsf{h}(\mathsf{x})]$ is a true sentence relative to $\mathscr{M}$ and a variable assignment g if and only if all $\mathsf{g}(\mathsf{x})$ make $[\mathsf{f}(\mathsf{x}) \rightarrow \mathsf{h}(\mathsf{x})]^{\mathscr{M},g}$ true.

Today's PredC originated with Frege (1879). It was complemented by Montague with the assignment g, lambda calculus, and the intension-extension dichotomy. In DBS, the assignment g is replaced by substitution variables with simultaneous substitution, the role of lambda reduction is taken by time-linear derivations in the speak and the hear mode, and instead of the intension-extension dichotomy (based on possible worlds) DBS uses the type-token relation, implemented as efficient computational pattern matching based on nonrecursive feature structures with ordered attributes, called proplets.The syntactic rules of PropC and their semantic interpretation rely on recursion.

For example, if $(p \wedge q)$ and r are sentences, then $((p \wedge q) \wedge r)$, $((p \wedge q) \vee r)$, $((p \wedge q) \rightarrow r)$, and $((p \wedge q) = r)$ as well as $(r \wedge (p \wedge q))$, $(r \vee (p \wedge q))$, $(r \rightarrow (p \wedge q))$, and $(r = (p \wedge q))$ are also sentences. The (i) order of the sub-sentences does not make a difference semantically (symmetry) except for the operator '$\rightarrow$' (asymmetry), the (ii) sentences resulting from the variations all have well-defined semantic interpretations relative to $\mathscr{M}$, and (iii) there is no upper bound on the number of variations because there is no limit on the length of PropC formulas.

2 PredC Overgeneration

It is similar for $\forall x[f(x) \rightarrow h(x)]$ and $\exists y[i(y) \wedge j(y)]$ of PredC, except that (i) the semantic interpretation relative to a model $\mathscr{M}$ requires the variable assignment g, (ii) functors may take more than one quantified variable as an argument, e.g., $f(x,y)$ or $f(x,y,z)$, and (iii) alternative orders of substitution may produce a systematic ambiguity such as the following:

2.1 ALLEGED AMBIGUITY OF Every dog finds a bone

1. $\forall x[dog(x) \rightarrow \exists y[bone(y) \wedge find(x,y)]]$
   Every dog finds a bone
2. $\exists y[bone(y) \wedge \forall x[dog(x) \rightarrow find(x, y)]]$
   There exists a bone which every dog finds

When PredC is used as the ready-made semantic interpretation of an independently motivated ("autonomous") syntax, specifically "innate" PSG, both readings are automatically assigned. Thereby reading (1) is an intuitively correct representation of the meaning but reading (2) is not. Without being a genuine paraphrase, the PrepC structure of reading (2) is an artifact of the formalism. Because the alleged ambiguity occurs whenever the $\exists$ and $\forall$ quantifiers appear in the same formula, it is a systematic overgeneration of PredC.

3 Determiners in DBS

The DBS alternative to the PredC formulas in 2.1 is a single set of proplets, connected by semantic relations, coded by address.

3.1 UNAMBIGUOUS DBS ANALYSIS OF Every dog finds a bone

$$
\begin{bmatrix} \text{sur:} \\ \text{noun: dog} \\ \text{cat: snp} \\ \text{sem: pl exh} \\ \text{fnc: find} \\ \dots \\ \text{prn: 4} \end{bmatrix}
\begin{bmatrix} \text{sur:} \\ \text{verb: find} \\ \text{cat: \#ns3 \#a decl} \\ \text{sem: pres} \\ \text{arg: dog bone} \\ \dots \\ \text{prn: 4} \end{bmatrix}
\begin{bmatrix} \text{sur:} \\ \text{noun: bone} \\ \text{cat: snp} \\ \text{sem: indef sg} \\ \text{fnc: find} \\ \dots \\ \text{prn: 4} \end{bmatrix}
$$

As in natural language, but unlike 2.1, there is no ambiguity, no coordination, and no coreference. The determiner aspect of $\forall$x representing every in 2.1 is coded as the features [cat: snp] (singular nounphrase) and [sem: pl exh] (plural exhaustive) of the *dog* proplet, while that of $\exists$y representing a(n) is coded as the features [cat: snp] (singular noun phrase) and [sem: sg indef] (singular indefinite) of the *bone* proplet. The semantic relations between the proplets *dog* and *find* are subject/predicate, and between *find* and *bone* object\predicate.

4 PredC Undergeneration

From a linguistic point of view, the counterpart to PredC overgeneration is PredC undergeneration, i.e., certain meaningful grammatical constructions cannot be properly expressed. The classic example is the *donkey sentence*[4]: Every farmer who owns a donkey beats it (Geach 1962). The PredC derivation is driven by systematic substitution and results in the following formula:

4.1 INCORRECT ANALYSIS OF WELL-FORMED SENTENCE IN PREDC

Every farmer who has a donkey beats it

$\forall x[[\text{farmer}(x) \wedge \exists y[\text{donkey}(y) \wedge \text{own}(x,y)]] \rightarrow \text{beat}(x,y)]$

The English sentence is grammatical and meaningful, and the substitutions of the derivation are correct, but the resulting formula is semantically inappropriate because the variable y in beat(x,y) is not in the scope of the quantifier $\exists$y binding donkey(y).[5]

5 Coreference by Address

Alternative to treating coreference by means of quantifiers binding variables, DBS treats all semantic relations of structure, including coreference (CC 6.5), by means of proplet-internal address values.

5.1 INTERPRETATION OF THE DONKEY SENTENCE IN DBS

noun: farmer	verb: own	noun: **donkey**	verb: beat	noun: **(donkey 17)**
cat: snp	cat: #n′ #a′ v	cat: snp	cat: #ns3′ #a′ decl	cat: snp
sem: pl exh	sem: pres	sem: indef sg	sem: pres	sem: sg
fnc: beat	arg: ∅ donkey	fnc: own	arg: farmer (donkey 17)	fnc: beat
mdr: (own 17)	mdd: (farmer 16)	mdr:	mdr:	mdr:
nc:	nc:	nc:	nc:	nc:
pc:	pc:	pc:	pc:	pc:
prn: 16	prn: 17	prn: 17	prn: 16	prn: 16

[4]The quantifier scope problem manifested by the donkey sentence was recognized in the middle ages (Walter Burley 1328/Gualterus Burlaeus 1988). It is one of several instances in which PredC can not provide the semantically correct quantifier scope. Discourse Representation Theory (DRT, Kamp 1980, Kamp and Reyle 1993) attempted to solve the problem of the donkey sentence while trying to maintain the sign-based substitution-driven foundation of PredC.

The grammatical embedding is reflected by the prn values, which are automatically incremented from 16 to 17 in the transition from the main to the sub-clause and decremented from 17 to 16 in the transition back to the main clause. The coreferential pronoun it in the main clause is represented by the extrapropositional address (donkey 17), which refers to the antecedent[6] in the sub-clause.

6 In PredC, Propositions Denote Truth Values

Another important difference between DBS and PredC is the respective treatment of concepts like dog, small, find, bone, and big. In PredC, concepts are treated as elementary mini-propositions which denote truth values relative to a set-theoretic model. Formally defined as functors which may differ in the number of arguments, e.g., f(x) vs. f(x,y), they are connected by the propositional operators of PropC and the quantifiers of PredC:

6.1 Noun, verb, and adj flattened into mini-propositions

the noun dog is interpreted as x is a dog and written as $dog'(x)$
the adj little is interpreted as x is little and written as $little'(x)$
the 1-place verb snore is interpreted as x snores and written as $snore'(x)$
the 2-place verb find is interpreted as x finds y and written as $find'(x, y)$
the 3-pl. verb give is interpreted as x gives y z and written as $give'(x, y, z)$

This allows to represent, for example, The little dog found a big bone as five mini-propositions which are coordinated with the propositional operator $\wedge$ and have the variables x and y bound by two $\exists$ quantifiers:

6.2 PredC representation of The little dog found a big bone

$$\exists x[dog'(x) \wedge little'(x) \wedge \exists y[bone'(y) \wedge big'(y) \wedge find'(x, y)]]$$

The meaning difference between the constants dog, little, bone, big, and find depends on the denotation function F and the assignment function g, provided they are defined by the logician, which is usually not the case. The reason may be shown by explicitly defining a possible model for a PredC formula:

6.3 Minimal model for the PredC formula 6.2

Let $\mathscr{M}$ be a model <A, B, F, g>, where A is an infinite set of objects or individuals, B a finite set of basic expressions, F a denotation function from B into A^*, and g an assignment function from variables into A^*.

[5]There have been numerous proposals to avoid the "dangling variable" by fronting the existential quantifier. In this way, the y in donkey(y) would get bound by $\exists$y but at the cost of losing compositionality (King and Lewis 2017), which is methodologically unacceptable.

[6]In linguistics, the term 'antecedent' is used not only for inferences, but also for the full noun referent, here a donkey, preceding a coreferential pronoun, here it (CLaTR 11).

For illustration, let us define A, B, F, and g as follows:

$A = \{a_1, a_2, a_3\}$
$B = \{\text{dog, small, big, bone, eat}\}$
$F(\text{dog}) = a_1$, $F(\text{small}) = \{a_1\}$, $F(\text{big}) = \{a_2\}$, $F(\text{bone}) = a_2$, $F(\text{eat}) = <a_1\ a_2>$,
$g(x) = a_1$, $g(y) = a_2$

Based on the definitions in 6.3, the formula 6.2 is well-formed and true in $\mathscr{M}$. However, if we defined F(dog) as a_3, for example, the formula would be false.

In summary, because PredC's pre-computational ontology provides neither a memory nor an interface component for autonomous recognition and action, formal models like 6.3 must be defined by hand, though a complete modeling of even the tiniest corner of reality (i) is out of reach and (ii) without any practical purpose. Therefore, explicit models are almost never defined. Instead logicians circumscribe models by means of conditionals which encode their intuitions about truth. DBS, in contrast, treats the world surrounding the agent's cognition *as given*[7] and limits itself to automatically monitor the agent's recognition and action computationally, using the agent's on-board interface and memory components.

7 In DBS, Propositions are Content

From a linguistic of view, treating nouns, adjs, and verbs uniformly as minipropositions (6.1) is semantically misguided because (i) it loses the classical distinction between referents, properties, and relations (CC 1.5.3), and (ii) obscures the empirical fact that properties and relations do not refer. Also, the use of coordination and coreference in connection with the quantifiers ∀ and ∃ (iii) violates the methodological standard of surface compositionality (FoCL 4.5) because there is neither coordination nor coreference in natural language expressions such as The little dog found a big bone.

DBS, in contrast, differentiates concepts into the three semantic kinds *referent, property*, and *relation*, with the syntactic correlates of elementary *noun, adj*, and *verb* (CC 1.5). Also, instead of propositions "denoting truth-values," as in PropC and PredC, propositions 'are content' in DBS:

7.1 THE CONTENT OF The little dog found a big bone

sur:	sur:	sur:	sur:	sur:
noun: **dog**	adj: **little**	verb: **find**	adj: **big**	noun: **bone**
cat: def sg	cat: adn	cat: #n′ #a′ decl	cat: adn	cat: def sg
sem: animal	sem: pad	sem: ind past	sem: pad	sem: dog food
fnc: **find**	mdd: **dog**	arg: **dog bone**	mdd: **bone**	fnc: **find**
mdr: **little**	mdr:	mdr:	mdr:	mdr: **big**
nc:	nc:	nc:	nc:	nc:
pc:	pc:	pc:	pc:	pc:
prn: 47	prn: 47	prn: 47	prn: 47	prn: 47

The proplets of a content are a set (order-free), held together by (i) a common prn value, here 47, and (ii) the semantic relations of structure coded by address, e.g., extrapropositional (table 47) or intrapropositional table (using the prn number of the proposition by default).

In addition to coding a semantic relation from one proplet to another, an address specifies a proplet's unique *storage location* (CC 12.4.4) in the agent's content-addressable A-memory (CC 2.3). The core value of the address is used for finding the token line (vertical) and the prn value for finding the position within the token line (horizontal). Computationally, the token line is found by the letter sequence using string search (Knuth et al. 1977) in combination with a trie structure; the position in a token line is found using hashing.

8 Extending PredC to Possible Worlds

To accommodate the classical modalities of necessity ($\Box$) and possibility ($\Diamond$) as well as change in time and space, Montague extended models like 6.3 into *model structures* by adding the infinite sets I for moments of time and J for possible worlds (reification):

8.1 DEFINITION OF A MODEL STRUCTURE

A model structure is defined as the sextuple <A, I, J, B, F, g>, where A is an infinite set of objects or individuals, I an infinite set of moments of time, J an infinite set of possible worlds, B a finite set of basic expressions, F a denotation function from B into A^*, and g an assignment function from variables into A^*. The elements of J are in linear order and I×J is the Cartesian product of I and J.

The operators of 1.1 (3, 4) are extended to $\Box$ (necessary) and $\Diamond$ (possible):

Syntax:
(i) If ϕ is a sentence, then $\Box\phi$ is a sentence.
(j) If ϕ is a sentence, then $\Diamond\phi$ is a sentence.

Semantics:
(i) If ϕ is a sentence, then $\Box\phi^{\mathscr{M},\, i\, j\, g}$ is 1 iff $\phi^{\mathscr{M},\, i',\, j',\, g}$ is 1 for all $<i', j'> \varepsilon\ I \times J$.
(j) If ϕ is a sentence, then $\Diamond^{\mathscr{M},\, i\, j\, g}$ is 1 iff $\phi^{\mathscr{M},\, i',\, j',\, g}$ is 1 for at least one $<i', j'> \varepsilon\ I \times J$.

This construct of possible world semantics raises the question: How can truth values characterize content? Let us consider an analogy from technology. On the computer screen, a black and white portrait is composed of two kinds of pixels, black and white, yet the pixel arrangement can be recognized as an individual face, and the more pixels the sharper the image. This is similar in a formal model with propositions which have only two denotations, $\emptyset$ and $\{\emptyset\}$.

Next compare a photo and a movie, both in black and white: they differ in that the pixels in a photo are static, as in the formal model 6.3, while the pixels in the movie change in time and location, as in the formal model structure 8.1. Like models,

[7] Brooks (1986): "The world is its own best model."

model structures rely on definitions produced by hand, resembling animation in film. TV and DBS, in contrast, may create content automatically, one by recording the agent-external reality, the other by monitoring cognition-external data.

TV and DBS differ in that (i) TV is limited to the output modalities of visual and auditory display, while DBS integrates a wide range of modalities in recognition as well as action, and (ii) TV is limited to displaying recorded images and sound, while DBS spontaneously processes (a) fresh input of raw data directly into content (recognition) and (b) currently activated content directly into raw data output (action), thus accommodating Hume's famous argument against postulating a *homunculus*[8] (Hume 1748). The technical basis is a content-addressable memory with a now front, regular clearance of the now front by leaving content behind in memory, automatic word form recognition and production, as well as operations for connecting input into content and for activating content by navigation, which are all absent in TV.

9 Semantic Relations of Structure in DBS

The proplets of (i) an elementary proposition and (ii) between the top verb proplets of coordinated propositions (TExer 2.1.6 ff.) as well as subordinated propositions (TExer 2.5, 2.6, 3.4–3.6) are connected by the four kinds of relation:

9.1 THE FOUR SEMANTIC RELATIONS OF STRUCTURE

1. subject/predicate
2. object\predicate
3. modifier|modified
4. conjunct−conjunct

Consider the following set of proplets of an intrapropositional content:

9.2 CONTENT OF Lucy found a big blue square .

sur: lucy	sur:	sur:	sur:	sur:
noun: [person x]	**verb: find**	adj: big	adj: blue	**noun: square**
cat: snp	cat: n′ a′ decl	cat: adn	cat: adnv	cat: sn
sem: nm f	sem: ind past	sem: pad	sem: pad	sem: sg
fnc: find	**arg: [person x] square**	mdd: square	mdd:	**fnc: find**
mdr:	mdr:	mdr:	mdr:	mdr: big
nc:	nc:	nc: blue	nc:	nc:
pc:	pc:	pc:	pc: big	pc:
prn: 14	prn: 14	prn: 14	prn: 14	prn: 14

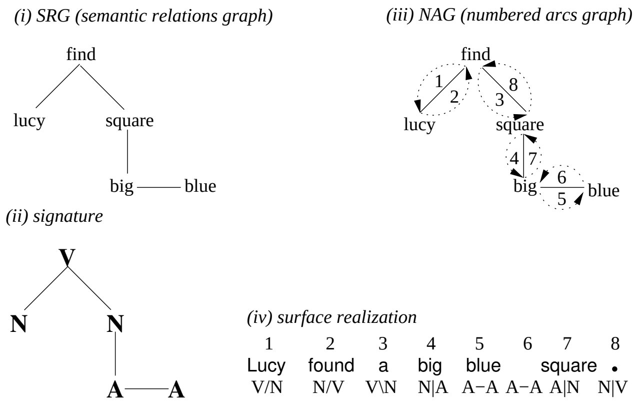
The subject/predicate relation is coded bidirectionally between (i) the core feature [noun: [person x]] of *lucy* and the continuation feature [arg: [person x]

[8]Rejecting a humunculus does not imply rejecting iconicity (similarity based on pattern matching), which is essential for computational similarity in DBS cognition (FoCL 3.3), pace Ogden&Richards (1923).

square] of *find* and (ii) the core feature [verb: find] of *find* and the continuation feature [fnc: find] of *lucy*. Similarly, the object\predicate relation is coded between (i) the core feature [noun: square] of *square* and the continuation feature [arg: [person x] square] of *find* and (ii) the core feature of [verb: find] of *find* and the continuation feature [fnc: find] of *square*. Correspondingly for *big* and *blue*.

By representing the semantic relations of structure with the operators /, \, |, and −, the content 9.2 may be shown graphically as follows:

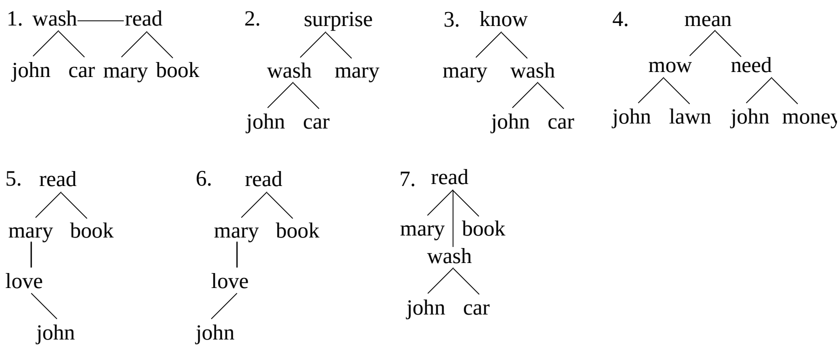
9.3 SEMANTIC RELATIONS OF STRUCTURE IN THE CONTENT 9.2



The graph is based on the proplets of 9.2. It characterizes the semantic relations of structure with four views. View (i), called the *semantic relations graph* (SRG), uses the core values lucy, find, square, big and blue as nodes. View (ii), called the *signature*, uses the core attributes N for noun, V for verb, and A for adj as nodes. View (iii), called the *numbered arcs graph* (NAG), supplements the SRG with numbered arcs which are used in (iv), called the *surface realization*, to show the navigation which activates content in the think mode and optionally realizes the language-dependent surfaces in a speak mode which rides piggyback on the think mode navigation. The concepts, as the elementary building blocks of DBS, are shown by placeholder values, using English base forms for convenience.

The operators /, \, |, and − are used also for extrapropositional relations:

9.4 RELATING TWO TRANSITIVE VERBS EXTRAPROPOSITIONALLY



1. conjunct−conjunct: John washed the car. Mary read a book.
2. subject/predicate: That John washed the car surprised Mary.
3. object\predicate: Mary knew that John had washed the car.
4. sbj/prd\obj: That John mows the lawn means that John needs money.[9]
5. adn mdr|mdd (subject gap): Mary who loves John read a book.
6. adn mdr|mdd (object gap): Mary who(m) John loves read a book.
7. adv modifier|modified: When John washed the car Mary read a book.

Which kind of relation may connect the predicates of two component propositions depends on the verb class. For example, 2 requires a psych verb (TExer 2.5), while 3 requires a mental state verb (TExer 2.6). Thus, it is impossible linguistically to use the same predicate[10] for constructing all seven constellations.

Once the semantic structure of a grammatical construction has been figured out graphically, writing the content as a set of proplets, e.g., 9.2, is easy. The subsequent steps are (a) the hear mode derivation using the lexical proplets provided by automatic word form recognition and (b) the think-speak mode derivation using the content derived in the hear mode (laboratory set-up). For details see TExer.

10 Properties Common to Hear, Think, and Think-Speak Operations

As an agent-based data-driven approach, DBS derivations require three kinds of operations: for (i) the hear mode, (ii) the think mode, and (iii) the think-speak mode. They have in common that they use a time-linear derivation order and share the following structural properties:

10.1 Structural properties common to all DBS operations

1. DBS operations consist of an antecedent, a connective, and a consequent.

2. The antecedent and the consequent are defined as sets of proplet patterns which are semantically connected by proplet-internal addresses.[11].

3. With the exception of inferences, the proplet patterns of the antecedent are the input pattern, while the proplet patterns of the consequent are the output pattern.

[9] Verbs which take a clausal subject and a clausal object simultaneously (class 4 in 9.4) include *entail, hint, imply, indicate, mean, presuppose,* and *suggest*. Thanks to Prof. Kiyong Lee for pointing it out.

[10] The criteria for establishing word classes are diverse (Levin 2009). For analyzing semantic relations of structure, DBS defines verb classes in terms of their possible valency filler (CLaTR 15).

[11] With the exception of suspension operations (11.1, 3) in the hear mode.

4. Inferences apply by matching input to the antecedent (deductive use, CC 3.5.1) or the consequent (abductive use, CC 3.5.2).

5. An operation is activated by content matching the input pattern (data-driven).

6. Binding the input constants to the variables of the input pattern enables the output pattern to derive the output.

7. The codomain of a variable in a pattern proplet may be restricted by an explicit list of possible values (variable restriction, e.g. CC 5.5.4).

The hear mode is the language variant of the agent's recognition. Language recognition, regardless of the medium (e.g., audition, vision) differs from nonlanguage recognition in that it provides an explicit processing order in the form of a left-associative surface sequence as input. The think mode controls the agent's actions. The operations of the think mode are of two kinds, (i) activation by navigation and (ii) inferencing. In the think-speak mode, either may be realized in the agent's natural language, technically based on lexicalization rules which sit in the sur slot of think mode operations.

Because the operations of the hear, think, and think-speak mode rely on computational pattern matching to apply, truly efficient pattern matching is of the essence in DBS. This requires that (i) the attributes of the pattern and the input proplet are in the same order and (ii) absence of recursion, as in the data structure of proplets. The opposite is absence of order and presence of recursion. Motivated by a notion of "generality"[12] inappropriate for computation, order-free recursive structure is uniquely ill-suited for efficient pattern matching, but standard in today's substitution-driven systems.

11 DBS Hear Operations

For building complex content in the hear mode of natural language communication, a next word (i.e., a lexical proplet provided by automatic word form recognition) is connected to the current sentence start by one of the following operation kinds:

11.1 THREE KINDS OF DBS HEAR MODE OPERATIONS

1. cross-copying (connective $\times$, as in SBJ$\times$PRD; TExer 6.3.1, 1)

2. absorption (connective $\cup$, as in DET$\cup$CN; TExer 6.3.1, 51)

3. suspension (connective $\sim$, as in ADV$\sim$NOM; TExer 6.3.1, 32)

[12] Carpenter (1992).

A hear mode operation takes two proplets as input and produces one or two proplets as output. The pattern matching is controlled by the syntactic category, regardless of the distinction between the semantic kinds concept, indexical, and name.

The following hear mode derivation of the content 9.2 uses operations of the kind (i) cross-copying and (ii) absorption. For (iii) suspension see TExer, 3.1.

11.2 TIME-LINEAR SURFACE-COMPOSITIONAL DERIVATION

Lucy	found	a	big	blue	square	.

automatic word form recognition

sur: Lucy	sur: found	sur: a	sur: big	sur: blue	sur: square	sur: .
noun: [person x]	verb: find	noun: n_1	adj: big	adj: blue	noun: square	verb: v_1
cat: snp	cat: n' a' v	cat: sn' snp	cat: adnv	cat: adnv	cat: sn	cat: v' decl
sem: nm f	sem: past	sem: indef sg	sem: pad	sem: pad	sem: sg	sem:
fnc:	arg:	fnc:	mdd:	mdd:	fnc:	arg:
mdr:	mdr:	mdr:	mdr:	mdr:	mdr:	mdr:
nc:	nc:	nc:	nc:	nc:	nc:	nc:
pc:	pc:	pc:	pc:	pc:	pc:	pc:
prn:	prn:	prn:	prn:	prn:	prn:	prn:

syntactic−semantic parsing

	sur: Lucy	sur: found	
1	noun: [person x]	verb: find	*cross−copying*
	cat: snp	cat: n' a' v	
	sem: nm f	sem: past	
	fnc:	arg:	
	mdr:	mdr:	
	nc:	nc:	
	pc:	pc:	
	prn: 14	prn:	

	sur: lucy	sur:	sur: a	
2	noun: [person x]	verb: find	noun: n_1	*cross−copying*
	cat: snp	cat: #n' a' v	cat: sn' snp	
	sem: nm f	sem: past ind	sem: indef sg	
	fnc: find	arg: [person x]	fnc:	
	mdr:	mdr:	mdr:	
	nc:	nc:	nc:	
	pc:	pc:	pc:	
	prn: 14	prn: 14	prn:	

	sur: lucy	sur:	sur:	sur: big	
3	noun: [person x]	verb: find	noun: n_1	adj: big	*cross−copying*
	cat: snp	cat: #n' #a' v	cat: sn' snp	cat: adnv	
	sem: nm f	sem: past ind	sem: indef sg	sem: pad	
	fnc: find	arg: [p. x] n_1	fnc: find	mdd:	
	mdr:	mdr:	mdr:	mdr:	
	nc:	nc:	nc:	nc:	
	pc:	pc:	pc:	pc:	
	prn: 14	prn: 14	prn: 14	prn:	

	sur: lucy	sur:	sur:	sur:	sur: blue	
4	noun: [person x]	verb: find	noun: n_1	adj: big	adj: blue	*cross−copying*
	cat: snp	cat: #n' #a' v	cat: sn' snp	cat: adnv	cat: sn	
	sem: nm f	sem: past ind	sem: indef sg	sem: pad	sem: sg	
	fnc: find	arg: [p. x] n_1	fnc: find	mdd: n_1	mdd:	
	mdr:	mdr:	mdr: big	mdr:	mdr:	
	nc:	nc:	nc:	nc:	nc:	
	pc:	pc:	pc:	pc:	pc:	
	prn: 14	prn: 14	prn: 14	prn: 14	prn:	

5	sur: lucy	sur:	sur:	sur:	sur:	sur: square	*absorption*
	noun: [person x]	verb: find	noun: n_1	adj: big	adj: blue	noun: square	*with*
	cat: snp	cat: #n' #a' v	cat: sn' snp	cat: adnv	cat: adnv	cat: sn	*simultaneous*
	sem: nm f	sem: past ind	sem: indef sg	sem: pad	sem: pad	sem: sg	*substitution*
	fnc: find	arg: [p. x] n_1	fnc: find	mdd: n_1	mdd:	fnc:	
	mdr:	mdr:	mdr: big	mdr:	mdr:	mdr:	
	nc:	nc:	nc:	nc: blue	nc:	nc:	
	pc:	pc:	pc:	pc:	pc: big	pc:	
	prn: 14	prn: 14	prn: 14	prn: 14	prn: 14	prn:	
6	sur: lucy	sur:	sur:	sur:	sur:	sur: •	*absorption*
	noun: [person x]	verb: find	noun: square	adj: big	adj: blue	verb: v_1	
	cat: snp	cat: #n' #a' v	cat: sn' snp	cat: adnv	cat: adnv	cat: v' decl	
	sem: nm f	sem: past ind	sem: indef sg	sem: pad	sem: pad	sem:	
	fnc: find	arg: [p. x] square	fnc: find	mdd: square	mdd:	arg:	
	mdr:	mdr:	mdr: big	mdr:	mdr:	mdr:	
	nc:	nc:	nc:	nc: blue	nc:	nc:	
	pc:	pc:	pc:	pc:	pc: big	pc:	
	prn: 14	prn: 14	prn: 14	prn: 14	prn: 14	prn:	

result

sur: lucy	sur:	sur:	sur:	sur:
noun: [person x]	verb: find	noun: square	adj: big	adj: blue
cat: snp	cat: #n' #a' decl	cat: sn' snp	cat: adnv	cat: adnv
sem: nm f	sem: past ind	sem: indef sg	sem: pad	sem: pad
fnc: find	arg: [p. x] square	fnc: find	mdd: square	mdd:
mdr:	mdr:	mdr: big	mdr:	mdr:
nc:	nc:	nc:	nc: blue	nc:
pc:	pc:	pc:	pc:	pc: big
prn: 14	prn: 14	prn: 14	prn: 14	prn: 14

The analysis is (i) *surface compositional* because each lexical item has a concrete sur value and there are no surfaces without a proplet analysis. The derivation order is (ii) *time-linear*, as shown by the stair-like addition of a next word proplet. The activation and application of operations is (iii) *data-driven* by automatic word form recognition.

The computational pattern matching of DBS operations is illustrated by the following application of a hear mode operation:

11.3 CROSS-COPYING *lucy* AND *find* WITH SBJ×PRD (line 1 in 11.2)

SBJ×PRD

pattern level	noun: α	verb: β	⇒	noun: α	verb: β
	cat: NP	cat: NP′ X v		cat: NP	cat: #NP′ X v
	fnc:	barg:		fnc: β	arg: α
	prn:K	prn:		prn: K	prn: K
		⇑			⇓
content level	sur: Lucy	sur: found		sur: lucy	sur:
	noun: **[person x]**	verb: **find**		noun: **[person x]**	verb: **find**
	cat: snp	cat: n′ a′ v		cat: snp	cat: #n′ a′ v
	sem: nm f	sem: ind past		sem: nm f	sem: ind past
	fnc:	arg:		fnc: **find**	arg: **[person x]**
	mdr:	mdr:		mdr:	mdr:
	nc:	nc:		nc:	nc:
	pc:	pc:		pc:	pc:
	prn: 17	prn: 17		prn: 17	prn: 17

Lexical lookup and syntactic-semantic concatenation are incrementally intertwined: the lookup of a new next word occurs only after the current next word has been processed into the current sentence start. In each concatenation, the language-dependent sur value provided by lexical lookup is omitted in the output, with the partial exception of names.

While a cross-copying operation like 11.3 produces two output proplets, an absorption operation produces only one. Consider the following example:

11.4 ABSORBING INTERPUNCTUATION INTO *find* WITH S∪IP (line 6)

$$
\begin{array}{llll}
 & \textbf{S}\cup\textbf{IP} & & \\
\textit{pattern level} &
\begin{bmatrix}\text{verb: } \beta \\ \text{cat: \#X}' \text{ VT} \\ \text{prn: K}\end{bmatrix} &
\begin{bmatrix}\text{verb: V\_}n \\ \text{cat: VT' SM}\end{bmatrix} \Rightarrow &
\begin{bmatrix}\text{verb: } \beta \\ \text{cat: \#X}' \text{ SM} \\ \text{prn: K}\end{bmatrix} \\
 & & \text{b} \Uparrow & \Downarrow \\
\textit{content level} &
\begin{bmatrix}\text{sur:} \\ \text{verb: find} \\ \text{cat: \#n}' \text{ \#a}' \ \mathbf{v} \\ \text{sem: ind past} \\ \text{arg: [person x] square} \\ \text{mdr:} \\ \text{nc:} \\ \text{pc:} \\ \text{prn: 17}\end{bmatrix} &
\begin{bmatrix}\text{sur: .} \\ \text{verb: v\_1} \\ \text{cat: } \mathbf{v}' \ \mathbf{decl} \\ \text{sem:} \\ \text{arg:} \\ \text{mdr:} \\ \text{nc:} \\ \text{pc:} \\ \text{prn:}\end{bmatrix} &
\begin{bmatrix}\text{sur:} \\ \text{verb: find} \\ \text{cat: \#b}' \text{ \#a}' \ \mathbf{decl} \\ \text{sem: ind past} \\ \text{arg: [person x] square} \\ \text{mdr:} \\ \text{nc:} \\ \text{pc:} \\ \text{prn: 17}\end{bmatrix}
\end{array}
$$

The names of all three kinds of DBS hear mode operations (11.1) have an input1 connective input2 structure, where input2 matches a next word and input1 looks for matching input in the sentence start, i.e., the set of proplets at the current now front.

12 Activation in the DBS Think and Think-Speak Modes

Of the two kinds of DBS think mode operations, navigation operations serve to activate existing content and consist of one input and one output pattern. Inference operations serve to derive new content from given content and consist of an open number of input and output patterns. Both kinds use a time-linear derivation order and both may optionally produce language-dependent surfaces relying on the same lex rules.

Intrapropositionally, navigational DBS think and think-speak mode operations alike activate content by traversing the semantic relations of structure in both directions. Accordingly, the intrapropositional operations traversing subject/predicate are V↙N and N↗V, those traversing object\predicate are V↘N and N↖V, those traversing the modifier|modified relation adnominally are N↓A and A↑N (and similarly for adverbial V|A), and those traversing the conjunct−conjunct relation N−N are N→N and N←N (and similarly for A−A and V−V).[13]

[13]For a more detailed account see TExer.

The same holds for extrapropositional activations, except that extrapropositional coordination is uni-directional in the direction of time and requires an inference for traversal in the anti-temporal direction. In the DBS graph analysis 9.3, which happens to be intrapropositional, the dual traversals are shown in the (iii) NAG (numbered arcs graph) and applied in the (iv) surface realization.

The following examples refer to the arc numbers in the NAG of 9.3:

12.1 Navigating with $\mathsf{V}\swarrow\mathsf{N}$ from *find* to *Lucy* (arc 1)

$$
\begin{array}{llcll}
 & \mathbf{V\swarrow N} & & & \\
\begin{array}{l}\textit{pattern}\\ \textit{level}\end{array} &
\begin{bmatrix}\text{verb: } \alpha \\ \text{arg: } \beta\ \text{X} \\ \text{prn: K}\end{bmatrix} & \Rightarrow &
\begin{bmatrix}\text{sur: lexnoun}(\hat{\beta}) \\ \text{noun: } \beta \\ \text{fnc: } \alpha \\ \text{prn: K}\end{bmatrix} & \text{\#-mark } \beta \text{ in the } \mathsf{arg} \text{ slot of proplet } \alpha. \\
 & \Uparrow & & \Downarrow & \\
\begin{array}{l}\textit{content}\\ \textit{level}\end{array} &
\begin{bmatrix}\text{sur:} \\ \text{verb: } \textbf{find} \\ \text{cat: \#n}'\ \text{\#a}'\ \text{decl} \\ \text{sem: ind past} \\ \text{arg: } \textbf{[person x]}\ \text{square} \\ \dots \\ \text{prn: 14}\end{bmatrix} & &
\begin{bmatrix}\text{sur: } \mathsf{Lucy} \\ \text{noun: } \textbf{[person x]} \\ \text{cat: snp} \\ \text{sem: nm f} \\ \text{fnc: } \textbf{find} \\ \dots \\ \text{prn: 14}\end{bmatrix} &
\end{array}
$$

In the think mode, the $\mathsf{lexnoun}(\hat{\beta})$ operation in the sur slot of the output proplet is switched off, but switched on in the think-speak mode. The variable $\hat{\beta}$ refers to a list which associates each core value with a language-dependent counterpart – with the exception of names, which are realized from a marker in the sur slot (CASM).

12.2 Navigating with $\mathsf{N}\nearrow\mathsf{V}$ from *Lucy* back to *find* (arc 2)

$$
\begin{array}{llcll}
 & \mathbf{N\nearrow V} & & & \\
\begin{array}{l}\textit{pattern}\\ \textit{level}\end{array} &
\begin{bmatrix}\text{noun: } \beta \\ \text{fnc: } \alpha\ \text{Y} \\ \text{mdr: Z} \\ \text{prn: K}\end{bmatrix} & \Rightarrow &
\begin{bmatrix}\text{sur: lexverb}(\hat{\alpha}) \\ \text{verb: } \alpha \\ \text{arg: \#}\beta\ \text{X} \\ \text{prn: K}\end{bmatrix} & \text{\#-mark } \alpha \text{ in the } \mathsf{fnc} \text{ slot of proplet } \beta. \\
 & \multicolumn{3}{l}{\text{Z is NIL, or elementary and \#-marked.}} & \\
 & \Uparrow & & \Downarrow & \\
\begin{array}{l}\textit{content}\\ \textit{level}\end{array} &
\begin{bmatrix}\text{sur: lucy} \\ \text{noun: } \textbf{[person x]} \\ \text{cat: snp} \\ \text{sem: nm f} \\ \text{fnc: } \textbf{find} \\ \dots \\ \text{prn: 14}\end{bmatrix} & &
\begin{bmatrix}\text{sur: } \mathsf{found} \\ \text{verb: } \textbf{find} \\ \text{cat: \#n}'\ \text{\#a}'\ \text{decl} \\ \text{sem: ind past} \\ \text{arg: } \textbf{\#[person x]}\ \text{square} \\ \dots \\ \text{prn: 14}\end{bmatrix} &
\end{array}
$$

Production of the surface found by $\mathsf{lexverb}(\hat{\alpha})$ is based on the features $\mathsf{[verb{:}\ find]}$, $\mathsf{[cat{:}\ \#n'\ decl]}$, and $\mathsf{[sem{:}\ ind\ past]}$ of the output proplet.

13 DBS Inferencing

For DBS inferencing, please see CC 3.4-3.6, 4, 5, and Part II.

14 Conclusion

The main difference between the semantics of DBS and PredC (Predicate Calculus) is the agent-based data-driven ontology of DBS vs. the sign-based substitution-driven ontology of PredC. As a consequence, DBS requires an on-board interface component and an on-board memory, while PredC does not. Also, a sign-based approach has no room for distinct hear and speak modes, while DBS treats them as the language part of recognition and action.

PredC is perhaps more general and parsimonious, but DBS avoids some long-standing problems of PredC, such as over- (2) and undergeneration (4), and a dependence of contingent meaning on hand-crafted models (6 and 8). Also, because DBS implements agent-based recognition and action, including the hear (11), think, and think-speak modes (12), it is better suited for building a talking autonomous robot than systems inherently without on-board interface and memory components.

Bibliography

Barwise, J., and J. Perry (1983) *Situations and Attitudes*, Cambridge, Mass.: MIT Press

Brooks, R.A. (1986) "A Robust Layered Control System for a Mobile Robot," *IEEE Journal of Robotics and Automation* 2.1:14–23

Carpenter, B. (1992) *The Logic of Typed Feature Structures*, Cambridge: CUP

CASM = Hausser, R. (2017) "A computational treatment of generalized reference," *Complex Adaptive Systems Modeling*, Vol. 5.1:1–26

CC = Hausser, R. (2019) *Computational Cognition, Integrated DBS Software Design for Data-Driven Cognitive Processing*, pp. i–xii, 1–237, lagrammar.net

CLaTR = Hausser R. (2011) *Computational Linguistics and Talking Robots; Processing Content in DBS*, pp. 286. Springer (preprint 2nd ed. at lagrammar.net

FoCL = Hausser, R. (1999/2001/2014) *Foundations of Computational Linguistics, Human–Computer Communication in Natural Language, 3rd ed.*, pp. 522, Springer

Frege, G. (1879) *Begriffsschrift. Eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*, Halle: L. Nebert

Geach, P. (1972) "A Program for Syntax," in D. Davidson and G. Harman (eds.) *Semantics of Natural Language*, Dordrecht: D. Reidel, pp. 483–497

Gualterus Burlaeus (1988) *De puritate artis logicae tractatus longior*, Hamburg: Felix Meiner Verlag

Hausser, R. (1980) "The Place of Pragmatics in Model-Theory," in Groenendijk, J.A.G., T.M.V. Janssen, and M.B.J. Stokhof (eds) (1980) *Formal Methods in the Study of Language*, University of Amsterdam: Mathematical Center Tracts 135

Hume, D. (1748) *An Enquiry Concerning Human Understanding*, reprinted in Taylor, R. (1974) *The Empiricists: Locke, Berkeley, Hume*, Garden City, New York: Anchor Books, Doubleday

Kamp, J.A.W. (1980) "A Theory of Truth and Semantic Representation," in J.A.G. Groenendijk et al. (eds.)

Kamp, J.A.W., and U. Reyle (1993) *From Discourse to Logic*, Parts 1 and 2, Dordrecht: Kluwer

King, J.C., and K.S. Lewis, "Anaphora," *The Stanford Encyclopedia of Philosophy (Summer 2017 Edition),* Edward N. Zalta (ed.), URL = https://plato.stanford.edu/archives/sum2017/entries/anaphora/

Knuth, D.E., J.H. Morris, and V.R. Pratt (1977) "Fast Pattern Matching in Strings," SIAM Journal of Computing Vol. 6.2:323–350

Levin, B. (2009) "Where Do Verb Classes Come From?" http://web.stanford.edu/bclevin/ghent09vclass.pdf

Ogden, C.K., and I.A. Richards (1923) *The Meaning of Meaning*, London: Routledge and Kegan Paul

Montague, R. (1973) "The Proper Treatment of Quantification in Ordinary English," in J. Hintikka, J. Moravcsik, P. Suppes (eds.) *Approaches to Natural Language*, Dordrecht, Reidel: 221–242

TExer = Hausser, R. (2020) *Twentyfour Exercises in Linguistic Analysis, DBS software design for the Hear and the Speak mode of a Talking Robot* (lagrammar.net)