

Roland Hausser

Ontology of Communication

Agent-Based Data-Driven or
Sign-Based Substitution-Driven?

sur: Lucy
noun: [person x]
cat: snp
sem: nm f
fnc: find
mdr:
nc:
pc:
prn: 23

sur: found
verb: find
cat: #n' #a' decl
sem: ind past
arg: [person x] square
mdr:
nc:
pc:
prn: 23

sur: big
adj: big
cat: adn
sem: pad
mdd: square
mdr:
nc: blue
pc:
prn: 23

sur: blue
adj: blue
cat: adn
sem: pad
mdd:
mdr:
nc:
pc: big
prn: 23

sur: square
noun: square
cat: snp
sem: indef sg
fnc: find
mdr: big
nc:
pc: big
prn: 23

Ontology of Communication

Roland Hausser

Ontology of Communication

Agent-Based Data-Driven or
Sign-Based Substitution-Driven?

 Springer

Roland Hausser
Abteilung für Computerlinguistik
Friedrich-Alexander-Universität Erlangen
Erlangen, Germany

ISBN 978-3-031-22738-7 ISBN 978-3-031-22739-4 (eBook)
<https://doi.org/10.1007/978-3-031-22739-4>

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Ontology of Communication

Agent-Based Data-Driven or Sign-Based Substitution-Driven?

ROLAND HAUSSER



Friedrich-Alexander-Universität Erlangen-Nürnberg (founded 1743)

河宇士

Pen name given to the author by Professor Inseok Yang
President of the Korean Society of Linguistics, Seoul 1982

Preface

The precomputational foundations of theoretical computer science in the 1930s, 40s, and 50s inherited the sign-based substitution-driven ontology from mathematics and symbolic logic. Continuing from Frege, Hilbert&Ackermann, and Russell, the work of Church, Gödel, Kleene, Post, Tarski, and Turing resulted in a rich harvest of undecidable or undecided problems, such as the Entscheidungsproblem, the $P =? NP$ problem, Gödel's incompleteness proofs, the halting problem of Turing machines, and Post's correspondence problem.

To these venerable achievements, Chomsky added the claim that natural language is undecidable as well. Based on a sign-based substitution-driven ontology, the proof¹ relies on the complexity hierarchy of *Phrase Structure Grammar* (PSG, Chomsky hierarchy) and the combination of a context-free base with a transformation component, making the system, called *Generative Grammar* (GG), recursively enumerable, i.e. undecidable.

GG derivations are all started by the same input, namely the single nonterminal S node (for Sentence or Start), working like the start button of a stand-alone algorithm. The intended output is the random generation of well-formed expressions of a natural language, using the recursive substitution of nonterminal symbols which are finally substituted by terminal symbols. Just as the motor of a car requires a skilled human driver with vision and manipulation to keep the car on the road, the stand-alone generation algorithm of GG requires a native speaker and a linguist to distinguish between grammatical and ungrammatical output.

Based on the derivation principle of computing possible substitutions, GG is “not intended”² as a model of the speaker-hearer. Instead, the goal is a “universal” characterization of the “human language ability.” This may have been misinterpreted as the implicit promise that understanding the universal human language ability would fundamentally facilitate computational language processing, despite GG's undecidability. Today, however, after more than half a century and a tremendous international effort, the promise remains unfulfilled.

Based on the different derivation principle of computing possible continuations, the goal of DBS is a computational realization of natural language communication, defined as the automated transfer of content between cognitive agents. The input to the

¹ Peters, S., and R. Ritchie (1973) “On the Generative Power of Transformational Grammar,” *Information and Control*, Vol. 18:483–501

² Chomsky 1965, p. 9.

DBS speak mode is a content and the output a language-dependent surface. Speak mode derivations are inherently unambiguous, but may provide a choice between paraphrases. The input to the hear mode is a language-dependent surface and the output a content. Hear mode derivations, in contrast, may have more than one reading (ambiguity), but each reading of an n word form surface requires exactly $n-1$ operation applications.

The differences in the input and the output of GG and DBS, respectively, require different derivations. The substitution-based derivations of GG are vertical top-down; there is no inherent distinction between the speak and the hear mode, and no upper limit on the number of substitution operations for the length of an *output*.

DBS, in contrast, distinguishes between the speak and the hear mode from the outset. The modes share the horizontal (left-associative) direction of derivation, but take opposite inputs and outputs. In either mode, the number of operations is a function of the length of an *input*.

Following from these structural differences, GG and DBS have orthogonal hierarchies of computational complexity. The classes of DBS are the C1, C2, C3, B, and A languages. The classes of PSG are the regular, context-free, context-sensitive, and unrestricted languages.

The C-languages of DBS are so-called because they are *constant* in that each navigation (speak mode) or concatenation (hear mode) operation may take only a finite number of primitive operations, i.e. below a grammar-dependent upper bound; the only way to raise the complexity of a C-language above linear is *recursive ambiguity*. The B languages are so-called because the number of steps in an operation is Bounded by the length of the input. The A languages are so-called because they comprise only and All recursive languages.

In summary, the claim that natural language is undecidable holds for sign-based substitution-driven GG, defined as a transformation component on top of a context-free base. Agent-based data-driven DBS, in contrast, would require recursive ambiguity for any complexity degree above linear. Recursive ambiguity, however, is absent in natural language. Consequently, the processing of natural language in DBS is of linear complexity, which is a precondition for general real time performance of a talking autonomous robot.

Status Quo

A basic choice in contemporary cognitive science is between a sign-based substitution-driven and an agent-based data-driven ontology. Most of current research is invested in the sign-based substitution-driven approach, which originated in mathematical logic (Skolem 1920, Post 1936). As an alternative, this book explores the agent-based data-driven approach of Database Semantics (AIJ 1989, TCS 1992).

Manuscript Production

The camera-ready copy was made by the author using the L^AT_EX software. Thanks to Josef Moser of LinuxHilfe for fixing bugs and building the name index.

Background

Database Semantics (DBS, AIJ'89) is an agent-based data-driven theory of how natural language communication works. The prototype resembles natural agents in that it assumes (i) real bodies out there in the real world (embodiment, MacWhinney 2008) and (ii) an agent-internal cognition which includes an *interface* component for elementary recognition and action, a *memory* component for storing continuous monitoring, and an *operations* component for building and processing content.

In language communication, DBS agents switch between the speak and the hear mode (turn-taking, Sacks et al. 1974, Schegloff 2007). The speak mode is driven by navigating along the semantic relations in a cognition-internal content (input) resulting in cognition-external raw data (output), e.g. sound waves or pixels, which have no meaning or grammatical properties whatsoever but may be measured by natural science. The hear mode is driven by the raw data (input) produced by the speaker resulting in cognition-internal content (output). For communication to be successful, the content encoded by the speaker into raw data and the content decoded by the hearer from those raw data must be the same (minimal requirement).

Contents are built in agent-internal cognition from the classical semantic kinds *concept*, *indexical*, and *name*, and connected with the classical semantic relations of *functor-argument* (Chapter 4) and *coordination* (Chapter 5). The interaction between the agent-internal cognition and the agent-external raw data is based on the computational Mechanisms of (i) type-token matching for concepts, (ii) pointing at values of the on-board orientation system for indexicals, and implicit or explicit (iii) baptism for named referents. The computational complexity of natural language communication in DBS is linear (TCS'92).

Contents

1. Introduction	1
1.1 Ontology	1
1.2 Computational Cognition	2
1.3 Agent-Based Data-Driven vs. Sign-Based Substitution-Driven	2
1.4 Reconciling the Hierarchical and the Linear	3
1.5 Speak Mode Converts Hierarchy Into Linear Surface	3
1.6 Hear Mode Re-Converts Linear Input Into Hierarchical Output	4
1.7 Derivation Order	6
1.8 Type Transparency	6
1.9 Four Kinds of Type-Token Relations	8
1.10 Conclusion	10
2. Laboratory Set-Up of Database Semantics	11
2.1 Early Times	11
2.2 Study of the Language Signs	12
2.3 Using Successful Communication for the Laboratory Set-Up	15
2.4 From Operational Implementation to Declarative Specification	17
2.5 Formal Fragments of Natural Language	19
2.6 Incremental Upscaling Cycles	20
2.7 Conclusion	20
3. Outline of DBS	21
3.1 Building Content in the Agent's Hear Mode	21
3.2 Storage and Retrieval of Content in the On-Board Memory	24
3.3 Speak Mode Riding Piggyback on the Think Mode	25
3.4 Component Structure of Cognition	27
3.5 Sensory Media, Processing Media, and Their Modalities	28
3.6 Reference as a Purely Cognitive Process	29
3.7 Grounding	32
3.8 Conclusion	33

4.	Software Mechanisms of the Content Kinds	35
4.1	Apparent Terminological Redundancy	35
4.2	Restriction of Figurative Use to Concepts	38
4.3	Additional Constraint on Figurative se	40
4.4	Declarative Specification of Concepts for Recognition	41
4.5	Declarative Specification of Concepts for Action	42
4.6	Indirect Grounding of Indexicals and Names	43
4.7	Conclusion	44
5.	Comparison of Coordination and Gapping	45
5.1	Coordination of Elementary Adnominals	45
5.2	Coordination of Phrasal Adnominal Modifiers	46
5.3	Coordination of Phrasal Adverbial Modifiers	48
5.4	Coordination of Elementary Nouns as Subject	49
5.5	Intra- and Extrapropositional Verb Coordination	49
5.6	Extrasentential Coordination	51
5.7	Quasi Coordination in Subject Gapping	52
5.8	Quasi Coordination in Predicate Gapping	54
5.9	Quasi Coordination in Object Gapping	55
5.10	Conclusion	56
6.	Are Iterating Slot-Filler Structures Universal?	57
6.1	Language and Thought	57
6.2	Slot-Filler Iteration	61
6.3	Marked Slot-Filler Repetition in Infinitives	61
6.4	Marked Slot-Filler Repetition in Object Clauses	64
6.5	Marked Slot-Filler Repetition in Adnominal Clauses	65
6.6	Unmarked Slot-Filler Iteration in Gapping Constructions	66
6.7	Long-Distance Dependency	70
6.8	Conclusion	71
7.	Computational Pragmatics	73
7.1	Four Kinds of Content in DBS	73
7.2	Coactivation Resulting in Resonating Content	75
7.3	Literal Pragmatics of Adjusting Perspective	77
7.4	Nonliteral Pragmatics of Syntactic Mood Adaptation	79
7.5	Nonliteral Pragmatics of Figurative Use	80
7.6	Conclusion	81
8.	Discontinuous Structures in DBS and PSG	83
8.1	The Time-Linear Structure of Natural Language	83
8.2	Constituent Structure Paradox of PSG	86
8.3	Suspension in Database Semantics	89

8.4	Discontinuity With and Without Suspension in DBS	94
8.5	Conclusion	96
9.	Classical Syllogisms as Computational Inferences	99
9.1	Logical vs. Common Sense Reasoning	99
9.2	Categorical Syllogisms	100
9.3	Modus Ponendo Ponens	104
9.4	Modus Tollendo Tollens	105
9.5	Modi BARBARA and CELARENT	107
9.6	Modi DARII and FERIO	110
9.7	Modi BAROCO and BOCARDO	113
9.8	Combining S- and C-Inferencing	116
9.9	Analogy	117
9.10	Conclusion	118
10.	Grounding of Concepts in Science	119
10.1	The Place of Concepts in a Content	119
10.2	Definition of Concepts at the Elementary, Phrasal, or Clausal Level? ..	122
10.3	Extending a Concept to Its Class	124
10.4	Language Communication	126
10.5	Combining Concepts Into Content	128
10.6	Language Surfaces and Meaning ₁ Concepts in Communication	129
10.7	Extero- and Interoception	130
10.8	Emotion	131
10.9	Conclusion	131
11.	Function Words	133
11.1	Introduction	133
11.2	Interpreting Determiner Noun Combination in Hear Mode	135
11.3	Producing Determiner Noun Combination in Speak Mode	137
11.4	Prepositional Phrases	138
11.5	Auxiliaries	140
11.6	Subordinating Conjunctions	143
11.7	Coordinating Conjunctions	145
11.8	Conclusion	146
12.	Language vs. Nonlanguage Cognition	147
12.1	Building Blocks and Relations of Cognition	147
12.2	Example of a Content	148
12.3	Content as Input to the Speak Mode	148
12.4	Content as Output of the Hear Mode	149
12.5	Nonlanguage Cognition Provides Place Holder Values	150
12.6	Function Word Absorbs Content Word	151

12.7 Type-Token Matching in Recognition and Action	152
12.8 Language Communication	154
12.9 Conclusion	156
13. Grammatical Disambiguation	157
13.1 Degrees of Computational Complexity	157
13.2 Orthogonal LAG and PSG Complexity Hierarchies	158
13.3 Comparing Explicitly Defined Examples in PSG and DBS	159
13.4 Sub-Hierarchy of C1, C2, and C3 Lags	162
13.5 Applying LAG to Natural Language	164
13.6 From LAG to the Hear Mode	166
13.7 From the Hear Mode to the Speak Mode	168
13.8 Incremental Lexical Lookup in the Hear Mode	169
13.9 Ambiguity in Natural Language	170
13.10 Language Dependence of Grammatical Disambiguation	172
13.11 Bach-Peters Sentence	172
13.12 Conclusion	173
14. Database Semantics vs. Predicate Calculus	175
14.1 Definition of Predicate Calculus	175
14.2 PredC Overgeneration	176
14.3 Determiners	177
14.4 PredC Undergeneration	178
14.5 Coreference by Address	178
14.6 In PredC, Propositions Denote Truth Values	179
14.7 In Database Semantics, Propositions Are Content	180
14.8 Extending PredC to Possible Worlds	181
14.9 Semantic Relations of Structure	182
14.10 Properties Common to Hear, Think, and Think-Speak Operations	184
14.11 Hear Mode Operations	185
14.12 Activation in the Think and Think-Speak Modes	188
14.13 Inferencing	189
14.14 Conclusion	190
15. Agent-Based Memory as an On-Board Database	191
15.1 Input-Output of Conventional Database vs. On-Board Memory	191
15.2 Data Structure and Operations in a Record-Based Database	192
15.3 Data Structure and Operations	193
15.4 The On-Board Orientation System (OBOS)	195
15.5 Loom-Like Clearance of the Now Front	196
15.6 Resonating Content 1: Coactivation by Similarity	197
15.7 Resonating Content 2: Coactivation by Token Line Intersection	197

15.8 Resonating Content 3: Coactivation by Continuation	200
15.9 Memory-Based Concatenation in Nonlanguage Recognition	201
15.10 Conclusion	205
16. David Hume’s ‘Causation’ in Database Semantics	207
16.1 Asymmetry in Natural Coordination	207
16.2 Cause and Effect	208
16.3 Necessary, Unnecessary, Sufficient, and Insufficient Causes	209
16.4 Hume’s Copy Principle	209
16.5 Reconstruction of Elementary Recognition and Action	209
16.6 Computational Reconstruction of Complex Content	212
16.7 From Individual Contents to a Content Class	213
16.8 Four Different Kinds of Content	214
16.9 Accommodating Scenarios in DBS	216
16.10 Conclusion	217
17. Concepts in Computational Cognition	219
17.1 Concept-Based Interpretation of Indexicals and Names	219
17.2 Concepts Grounded in Science	220
17.3 ‘Natural Categories’ as Concepts	222
17.4 Technical Concepts as a Subclass of ‘Natural Categories’	223
17.5 Grammatical Categories	223
17.6 Hear Mode: Concatenating Proplets into Complex Content	224
17.7 Speak Mode: Linearization of a Content by Navigation	225
17.8 Natural Language Communication in Speech and Writing	226
17.9 Conclusion	228
18. Paraphrase and Ambiguity	229
18.1 Introduction: the Structure of Content	229
18.2 Speak Mode Paraphrase: Different Surfaces for Same Content	231
18.3 DBS Formalism for the Speak Mode (Language Production)	232
18.4 Hear Mode Ambiguity: Different Contents for Same Surface	234
18.5 Ambiguity is Language-Dependent	235
18.6 Grammatical Analysis of Ambiguity	236
18.7 Local vs. Global Ambiguities	237
18.8 Iterating Local Ambiguities	237
18.9 Conclusion	238
19. Recursion and Grammatical Disambiguation	239
19.1 Speak Mode in Database Semantics	239
19.2 Hear Mode in Database Semantics	240
19.3 Recursion	241
19.4 Conclusion	244

Name Index245

Bibliography249

1. Introduction



For long-term incremental upscaling to be successful, the computational reconstruction of a natural mechanism must be *input-output equivalent* with the prototype, i.e., the reconstruction must take the same input and produce the same output in the same processing order as the original. Accordingly, our computational reconstruction of natural language communication uses a time-linear derivation order for the speaker's output and the hearer's input. The surfaces serving as the vehicle of content transfer from speaker to hearer are raw data, e.g., sound waves or pixels, without meaning or any grammatical properties whatsoever, but measurable by natural science.¹

1.1 Ontology

The ontology of a field of science comprises the basic elements and relations assumed to allow a complete analysis of its phenomena. For example, the Presocratics tried to explain nature based on an ontology of fire, water, air, and earth. Today, the ontology of physics is based on a space-time continuum, protons, electrons, neutrons, quarks, neutrinos, etc.

Similarly in theories of meaning in philosophy. There was a time in which meaning was based on naming; for example, the celestial body rising in the morning and setting in the evening served as the meaning of the word sun. Then meaning became defined in terms of set-theoretic denotations in possible worlds. Which ontology is required for building the computational cognition of a talking robot?

Just as an ontology without subatomic particles is unsuitable for modern physics, an ontology of computational cognition without an agent, without a distinction between an agent-external reality and agent-internal processing, without interfaces for recognition and action, without a distinction between the speak and the hear mode, without an on-board memory (database), without an on-board orientation system (OBOS), and without an algorithm for moment-by-moment monitoring is unsuitable for the task of building a talking robot.

¹ Thanks to Prof. MacNeilage, director of the Phonetics Lab at UT Austin during the author's visit at the Linguistics Department as a Ph.D. student (1970–1974), for the opportunity to participate as a test person in phonetic research experiments. This instilled a permanent appreciation of raw data in language communication.

1.2 Computational Cognition

The ontological requirements for computational cognition were essentially laid down in the year 1945 as the von Neumann machine (vNm): the interface component of DBS (AIJ'01) corresponds to the vNm input-output device, the DBS on-board database corresponds to the vNm memory, and the DBS left-associative operations algorithm corresponds to the vNm arithmetic-logic.

Designing and building the computational cognition of a talking autonomous robot is not only of interest for a wide range of practical applications, but constitutes the ultimate standard for evaluating the many competing theories of natural language in today's linguistics, language philosophy, language psychology, and computer science. It leads from the sign-based substitution-driven ontology of mathematics and symbolic logic to the new (or extended) ontology of agent-based data-driven robotics in general and DBS in particular. It also leads from Generative Grammar (GG) and its attempt to discover an innate human language ability to the effective transfer of content from the speaker to the hearer by means of raw data.

Communication is successful if the content encoded by the speaker into raw data equals the content decoded from the raw data by the hearer. DBS constructs content from the three basic content kinds of (i) concept, (ii) indexical, and (iii) name. Each has its characteristic computational mechanism: concepts use computational pattern matching based on the type-token relation, indexicals use pointing at values of the agent's on-board orientation system, and names use an explicit or implicit act of baptism which inserts a named referent as core value into a name proplet (CASM'17).

1.3 Agent-Based Data-Driven vs. Sign-Based Substitution-Driven

Most analyses of natural language in today's linguistics, philosophy, and computer science rely on a precomputational, sign-based, substitution-driven ontology. Sign-based means: no distinction between the speak and the hear mode. Substitution-driven means: using a single start symbol as input for randomly generating infinitely many different outputs, based on possible substitutions by rewrite rules. Thereby different outputs are assigned the same denotation, i.e., True or False.

However, for a functionally complete, scientific reconstruction of natural language communication, the start button is uniquely unsuitable as input to the speak mode and the truth-values are uniquely unsuitable as output of the hear mode. In DBS, propositions do not denote but *are content*, and different propositions are different contents. A content is defined as a set of proplets, i.e., order-free (which is essential for storage in and retrieval from a content-addressable on-board database). Proplets are defined as nonrecursive feature structures with ordered attributes (which is essential for efficient pattern matching). The proplets in a content are connected by the classical semantic relations of structure, i.e., functor-argument and coordination, coded by address.

The ontology of DBS is agent-based and data-driven. Agent-based means: design of a cognitive agent with (i) an interface component for converting raw data into cog-

nitive content (recognition) and converting cognitive content into raw data (action), (ii) an on-board, content-addressable memory (database) for the storage and retrieval of content, and (iii) separate treatments of the speak- and the hear-mode. Data-driven means: (a) mapping a cognitive content as input to the speak mode into a language-dependent surface as output, and (b) mapping a surface as input to the hear mode into a cognitive content as output.

1.4 Reconciling the Hierarchical and the Linear

Content serving as input to the speak mode and as output of the hear mode is defined as a set of proplets, connected by the semantic relations of structure, coded by address:

1.4.1 CONTENT OF I saw you.

[sur: I noun: pro1 cat: s1 sem: sg fnc: see mdr: nc: pc: prn: 3]	[sur: saw verb: see cat: #n #a decl sem: past arg: pro1 pro2 mdr: nc: pc: prn: 3]	[sur: you noun: pro2 cat: sp2 sem: sg fnc: see mdr: nc: pc: prn: 3]
---	--	--

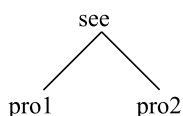
The classical semantic relations of structure are subject/predicate, object\predicate, modifier|modified, and conjunct–conjunct. The semantic relations in 1.4.1, are subject/predicate and object\predicate, indicated by **bold face** font. In order for communication to be successful, the input content of the speak mode and the output content of the hear mode must be the same (minimal condition).

1.5 Speak Mode Converts Hierarchy Into Linear Surface

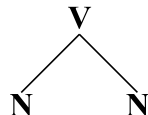
The speak mode converts the hierarchy of the input content into the linear structure of the output surface by navigating along the semantic relations of structure:

1.5.1 GRAPH ANALYSIS UNDERLYING PRODUCTION OF 1.4.1

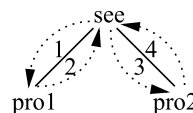
(i) SRG (semantic relations graph)



(ii) signature



(iii) NAG (numbered arcs graph)



(iv) surface realization

1	2	3	4
I	saw	you	.
V/N	N/V	V/N	NV

The (iv) surface realization consists of three lines, showing (1) the arc numbers, (2) the surfaces realized from the goal proplet, and (3) the traversal operations.

The operations driving the navigation in 1.5.1 are listed as follows:

1.5.2 SEQUENCE OF OPERATION NAMES AND SURFACE REALIZATIONS

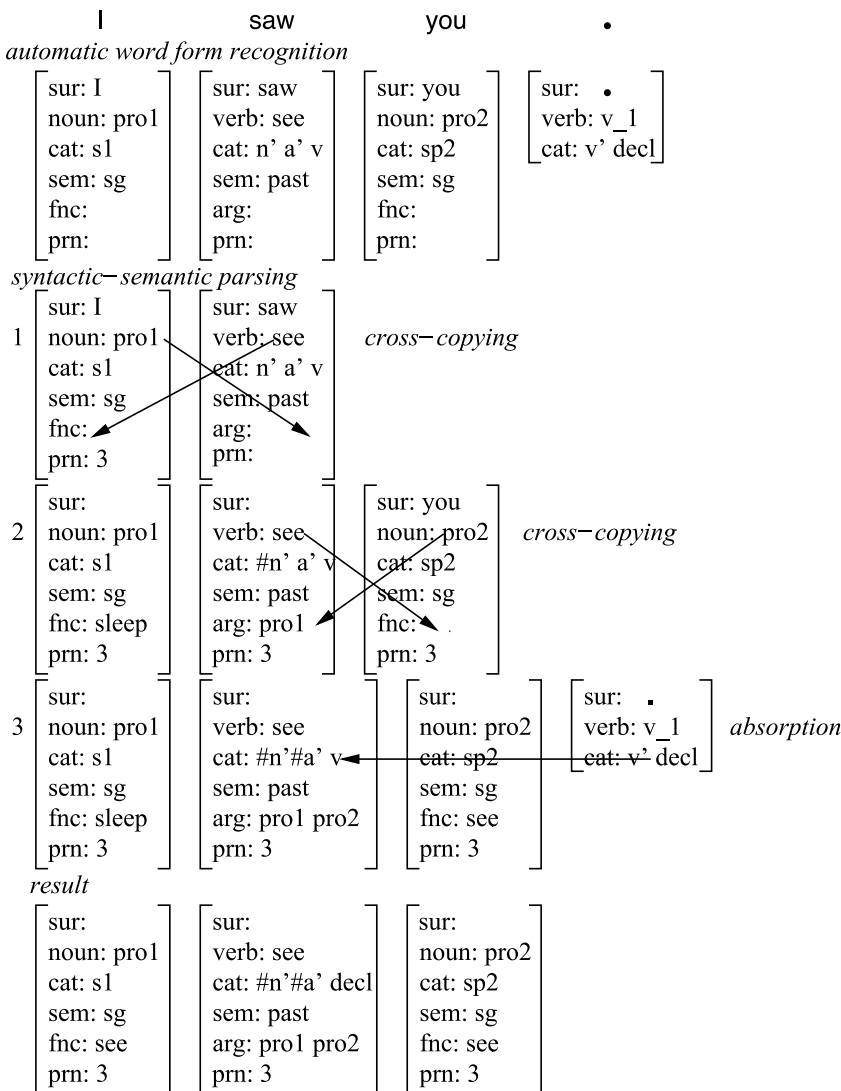
- arc 1: $V \setminus N$ from *see* to *pro1* | (TExer 2.3.8)
- arc 2: $N \setminus V$ from *pro1* to *see* saw (TExer 2.3.9)
- arc 3: $V \setminus N$ from *see* to *pro2* you (TExer 2.3.10)
- arc 4: $N \setminus V$ from *pro2* to *see* . (TExer 2.3.11)

1.6 Hear Mode Re-Converts Linear Input Into Hierarchical Output

The hear mode re-converts the stream of raw input data into the hierarchical structure of 1.4.1 by incremental lexical lookup and syntactic-semantic composition.

1.6.1 GRAPHICAL HEAR MODE DERIVATION OF THE CONTENT 1.4.1

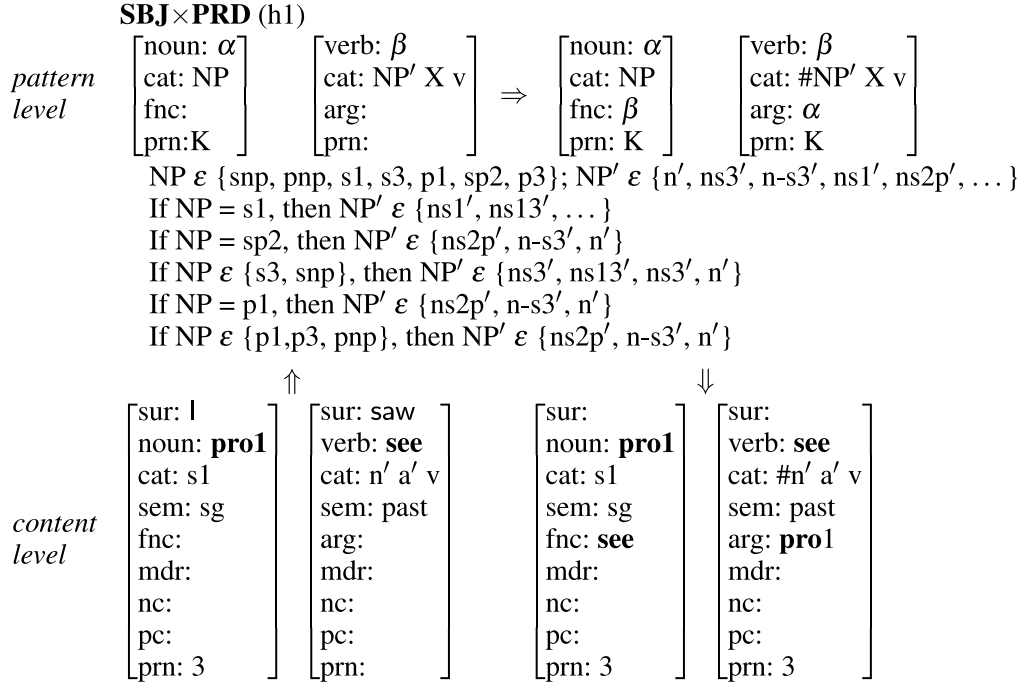
unanalyzed surface



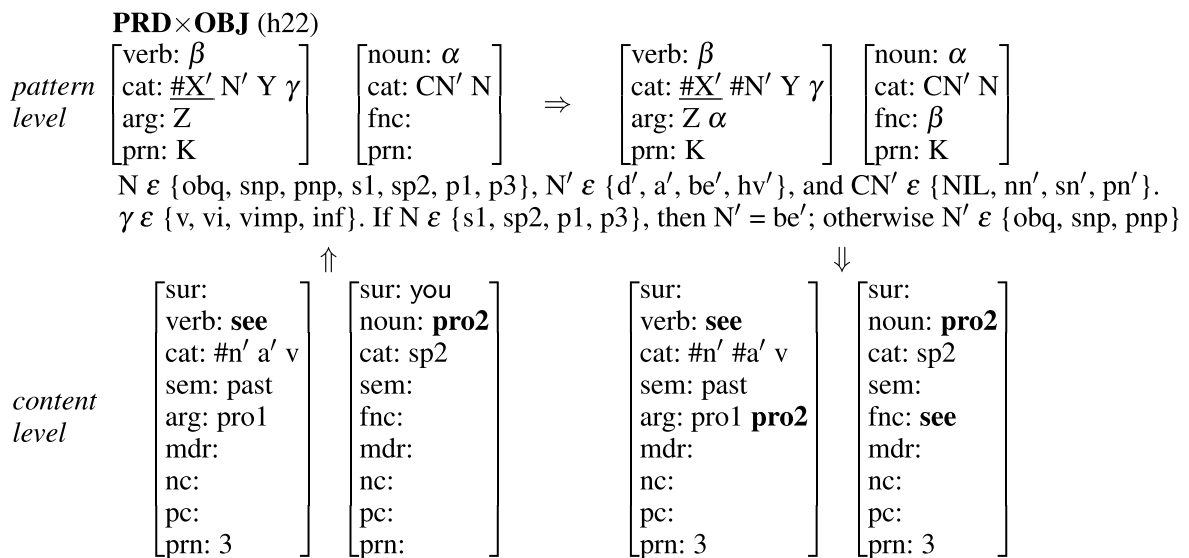
The composition is time-linear in that the current next word (lexical proplet) is related semantically to a proplet in the current sentence start (set of proplets already connected, at least partially).

The hear mode operations are of three kinds: (i) cross-copying (connective \times), (ii) absorption (connective \cup), and (iii) suspension (connective \sim). Operations with the same connective may re-introduce different semantic relations of structure, for example, $\text{SBJ} \times \text{PRD}$ and $\text{OBJ} \times \text{PRD}$, defined as follows:

1.6.2 CROSS-COPYING *pro1* AND *saw* WITH $\text{SBJ} \times \text{PRD}$ (LINE 1)



1.6.3 CROSS-COPYING *saw* AND *pro2* WITH $\text{PRD} \times \text{OBJ}^2$ (LINE 2)



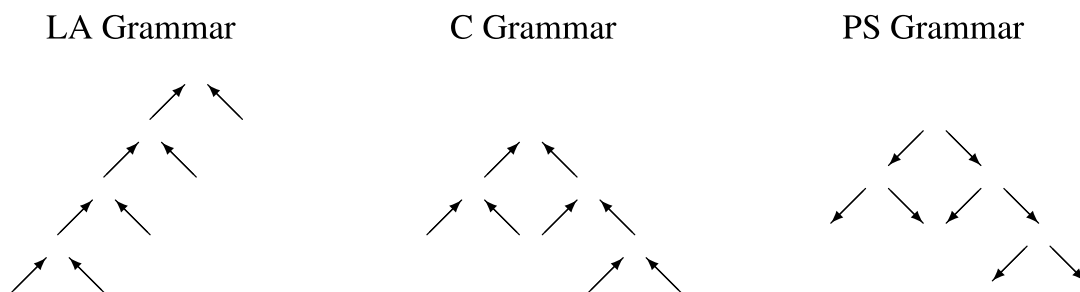
Comparison of the $\text{SBJ} \times \text{PRD}$ and the $\text{OBJ} \times \text{PRD}$ application illustrates the highly precise coding of grammatical detail, provided by the computational pattern matching

of DBS. For the complete declarative analysis of I saw you. in the speak and hear mode see TExer 2.3.

1.7 Derivation Order

The regular total-order derivation of time-linear Left-Associative Grammar (LAG) as the precursor of Database Semantics (DBS) differs from the irregular, partial-order derivations of Categorical Grammar (CG, bottom up) and Phrase Structure Grammar (PSG, top down):

1.7.1 THREE CONCEPTUAL DERIVATION ORDERS (FOCL 10.1.1)



bottom-up left-associative bottom-up amalgamating top-down expanding

The initial empirical test of using the left-associative derivation order for the syntactic-semantic analysis of a nontrivial set of natural language expressions was programming the time-linear derivations of 221 constructions of German and 114 constructions of English during a research stay at CSLI Stanford in 1984-1986.³

1.8 Type Transparency

The purpose of formal grammars for fragments of natural language is (i) a linguistically well-motivated analysis of examples which is suitable (ii) for efficient automatic derivation by a computer program and (iii) for systematic upscaling. This requires *input-output equivalence* between the declarative derivation order of the formal grammar and the procedural derivation order of the parser.

Called *type transparency* by Berwick and Weinberg (1984, p. 39), input-output equivalence between a formal grammar and its parser was originally intended also in PSG:

Miller and Chomsky's original (1963) suggestion is really that grammars be realized more or less directly as parsing algorithms. We might take this as a methodological principle. In this case we impose the condition that the logical organization of rules and structures incorporated in the grammar be mirrored rather exactly in the organization of the parsing mechanism. We will call this as follows:

1.8.1 DEFINITION OF ABSOLUTE TYPE TRANSPARENCY

- For any given language, parser and generator use the *same* formal grammar,
- apply the rules of the grammar *directly*,
- in the same *order* as the grammatical derivation,
- take the same *input* expressions as the grammar, and
- produce the same *output* expressions as the grammar.

For Phrase Structure Grammar (PSG), type-transparency is impossible. PSG is based on Post's (1936) production or rewrite systems, which were designed to mathematically characterize the notion of *effective computability* in recursion theory.⁴ In this original application, a derivation order based on the substitution of signs by other signs is perfectly natural. When Chomsky (1957) 'borrowed' the Post production system under the name Phrase Structure Grammar for analyzing natural language, he inadvertently inherited the substitution-driven derivation order.

Because rewrite systems take the start symbol as input, but a parser takes terminal strings, rewrite systems and their parsers are not input-output equivalent – which means that a type transparent PSG parser can not exist. Instead, huge intermediate structures are required to reconcile the time-linear input order of the parser and the top-down substitution order of the grammar's rewrite rules (Early Parser⁵, CYK Parser⁶, Tomita Parser⁷).

Consequently, (i) the computational complexity of PSG is polynomial,⁸ and (ii) debugging and upscaling in PSG-based parsing is greatly impeded: if a well-formed input is rejected or an ill-formed input accepted, the error must be found in the complex intermediate structures of the context-free PSG parser, which are not easy to read.

LA-grammar is not type-transparent either because the speak and the hear mode use different algorithms for different kinds of input, but the output of the speak mode is the input to the hear mode. In the hear mode, an error is located in the output close to where the time-linear derivation broke off or the ill-formed continuation began. Moreover, the error is explicitly documented in the automatic analysis serving simultaneously as the trace of the parse and the grammatical analysis.⁹

³ Thanks to CSLI Stanford for their generous hospitality, especially by providing the at the time most advanced workstations by HP with a team of helpful operators, and to the DFG for a five year Heisenberg grant. The research stay was initially intended to program the Montague Grammar defined in *Surface Compositional Grammar* (SCG). Even though the syntactic-semantic λ -derivations of surfaces into formulas of intensional logic were explicitly defined to high standard, a reasonable programming of the 'fragment' presented unsurmountable difficulties. In response, a time-linear approach was developed, programmed, and published as NEWCAT, including the source code written in Lisp.

⁴ See for example Church (1956), p. 52, footnote 119.

⁵ Early 1970.

⁶ Cocke and Schwartz 1970, Younger 1967

⁷ Tomita 1986

⁸ In contrast to the linear time complexity of type-transparent LAG/DBS (TCS'92).

⁹ FoCL 9.4, 10.4, 10.5, specifically 10.5.5.

1.9 Four Kinds of Type-Token Relations

The interaction between the DBS agent's computational cognition and its cognition-external surroundings is based on the pattern-matching of concepts. Recognition is a concept type matching raw data, resulting in a token stored in short term memory. Action is adapting a type to a purpose, resulting in a token realized as raw data.

DBS uses the type-token relation directly for elementary proplets of the semantic kinds concept, but indirectly for indexicals and names, and for complex contents of declarative, interrogative, and imperative sentences.

1.9.1 TYPE AND TOKEN OF A CONCEPT

<i>type</i>	<i>token</i>
sur: Hund noun: dog cat: def sg sem: fnc: mdr: nc: pc: prn:	sur: Hund noun: dog cat: def sg sem: fnc: snore mdr: nc: pc: prn: 24

The attributes fnc and prn of the type have no value, while those of the token have the values snore and 24. The sur value is from German.

1.9.2 TYPE AND TOKEN OF AN INDEXICAL

<i>type</i>	<i>token</i>	STAR
sur: you noun: pro2 cat: sp2 sem: fnc: mdr: nc: pc: prn:	sur: you noun: pro2 cat: sp2 sem: fnc: see mdr: nc: pc: prn: 24	S: veranda T: Monday A: John R: Mary prn: 24

The type has no prn value and no STAR to point at, while the token has the prn value 24 and may point at the STAR value John or Mary, depending on the syntax.

1.9.3 TYPE AND TOKEN OF A NAME

<i>type</i>	<i>token</i>
sur: Fido noun: cat: snp sem: m sg fnc: mdr: nc: pc: prn:	sur: Fido noun: [dog x] cat: snp sem: m sg fnc: mdr: nc: pc: prn:24

The type has no prn value and the core attribute noun has no ‘named referent’, while the token has the prn value 24 and the core attribute has the named referent [dog x].

Finally consider the type and token of a DBS proposition,¹⁰ defined as a content. The syntactic mood is specified by the verb’s cat value decl as a declarative.

1.9.4 TYPE OF A CONTENT

<i>type</i>	<i>type</i>	<i>type</i>
sur: noun: dog cat: snp sem: def sg fnc: find mdr: nc: pc: prn: K	sur: verb: find cat: #n' #a' decl sem: past ind arg: dog bone mdr: nc: pc: prn: K	sur: noun: bone cat: snp sem: indef sg fnc: find mdr: nc: pc: prn: K

This content is a type because there is no STAR and the prn value is a variable, here K. It is a nonlanguage content because the sur slots are empty.

1.9.5 CORRESPONDING TOKEN

<i>token</i>	<i>token</i>	<i>token</i>	<i>token</i>
sur: noun: dog cat: snp sem: def sg fnc: find mdr: nc: pc: prn: 12	sur: verb: find cat: #n' #a' decl sem: past ind arg: dog bone mdr: nc: pc: prn: 12	sur: noun: bone cat: snp sem: indef sg fnc: find mdr: nc: pc: prn: 12	S: yard T: friday A: sylvester R: 3rd: prn: 12

This content is a token because the three content proplets and the STAR proplet are connected by a common prn constant, here 12. According to the STAR, the content resulted as an observation by the agent Sylvester on Friday in the yard.

¹⁰ An elementary proposition is a content which uses exactly one prn value.

In summary, the types of individual proplets are lexical word form analyses which are provided by the on-board memory for automatic word form recognition/production. The type of a complex content results from concatenating proplet types with the semantic relations of structure. The content type of a proposition is turned into a content token by adding a STAR and replacing the prn variables with constants (simultaneous substitution).

1.10 Conclusion

In computer science, the input-output distinction holds (i) between a system and its external environment and (ii) between interacting components within a system. The sign-based substitution-driven ontology of Phrase Structure Grammar (PSG) avoids input-output interaction with the system-external reality by using the same S node like a start button as input for the random generation of all the different grammatical structures in the fragment. The agent-based data-driven ontology of DBS, in contrast, provides external nonlanguage input-output in (i) action and (ii) recognition between agents and their environment, and external language input and output between agents in the (iii) speak and (iv) hear modes.

The input to the (iii) speak mode is a hierarchical content and the output a linear surface. The input to the (iv) hear mode is a linear surface and the output a hierarchical content. The challenge for a functionally complete, scientific computational reconstruction of natural language communication is a bidirectional conversion between a linear and a hierarchical coding of the semantic relations of structure.

In DBS, the speak mode turns hierarchical input contents into linear output surfaces by *navigating* along the semantic relations of structure in the input. The hear mode turns linear input surfaces into hierarchical output content by incremental time-linear *syntactic-semantic composition* between the sentence start, defined as a set of proplets already connected (at least partially), and the next word, re-introducing the classical semantic relations of subject/predicate, object\predicate, modifier|modified, and conjunct–conjunct, coded by address.