1. Review of Current Framework

1. Types

1. Primitive Types

D	uninflected definite-noun-phrase
D_1, D_2, D_3, D_4, D_5, D_6	inflected definite-noun-phrase
C	common-noun-phrase
S	sentence

2. Derivative Types (functor-types)

if $\mathfrak{I}_1$ and $\mathfrak{I}_2$ are types, then so is $(\mathfrak{I}_1 \rightarrow \mathfrak{I}_2)$

3. Extremal Clause

nothing is a type except in virtue of clauses 1 and 2.

2. Composition Rule (original)

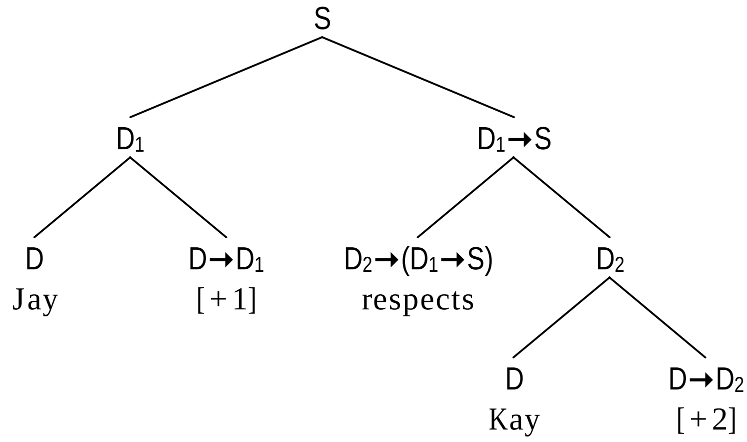
Every syntactic composition is achieved by applying a **unary functor** to an argument. In particular, combining a functor of type $\mathfrak{I}_1 \rightarrow \mathfrak{I}_2$ with a phrase of types $\mathfrak{I}_1$ results in a phrase of type $\mathfrak{I}_2$.

3. Example Types

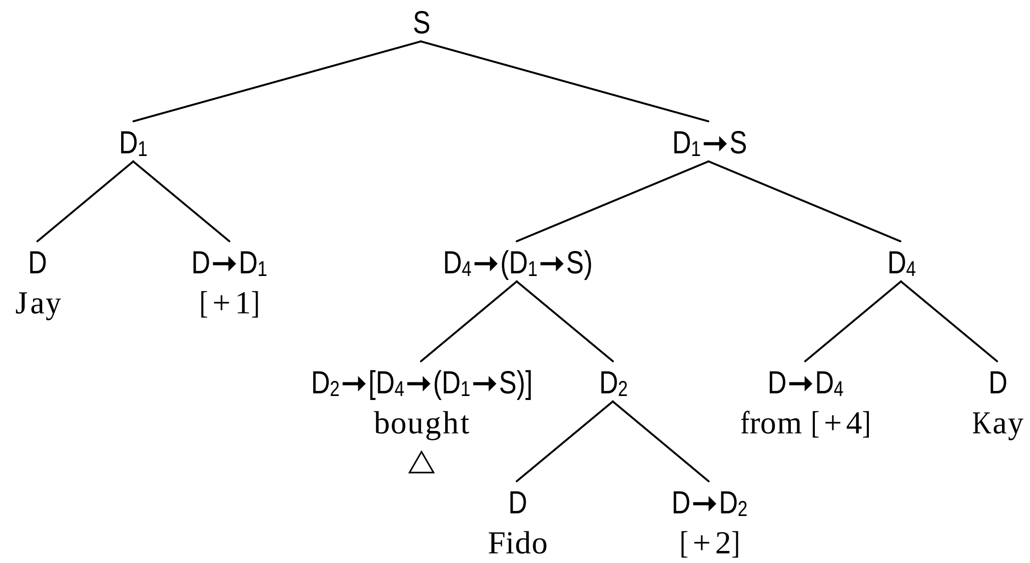
	Morpheme	Category	Type
(1)	[+1]	nominative-case marker	$D \rightarrow D_1$
(2)	[+2]	accusative-case marker	$D \rightarrow D_2$
(3)	to	dative-case marker	$D \rightarrow D_3$
(4)	from	ablative-case marker	$D \rightarrow D_4$
(5)	by	bylative-case marker	$D \rightarrow D_5$
(6)	of	genitive-case marker	$D \rightarrow D_6$
(7)	's		
(9)	not	1-place connective	$S \rightarrow S$
(10)	and, but	2-place connective	$S \rightarrow (S \rightarrow S)$
(11)	every, some, no	quantifier	$C \rightarrow [(D \rightarrow S) \rightarrow S]$
(12)	the	definite-determiner	$C \rightarrow D$
(14)	be	copula	$C \rightarrow (D_1 \rightarrow S)$
		(quasi-) transitive verb	$D_1 \rightarrow (D_1 \rightarrow S)$
(15)	who	(uninflected) relative-clause functor	$(D \rightarrow S) \rightarrow (C \rightarrow C)$
(19)	woman, man, dog, one	common-noun	C
(20)	mother, father	definite genitive-noun	$D_6 \rightarrow D$
(21)	friend, brother	indefinite genitive-noun	$D_6 \rightarrow C$
(22)	virtuous, happy	bare-adjective	C
		mod-adjective	$C \rightarrow C$
(23)	respect, admire	transitive verb	$D_2 \rightarrow (D_1 \rightarrow S)$
(24)	sell	di-transitive verb	$D_2 \rightarrow [D_3 \rightarrow (D_1 \rightarrow S)]$
(25)	buy	di-transitive verb	$D_2 \rightarrow [D_4 \rightarrow (D_1 \rightarrow S)]$

4. Example Analyses

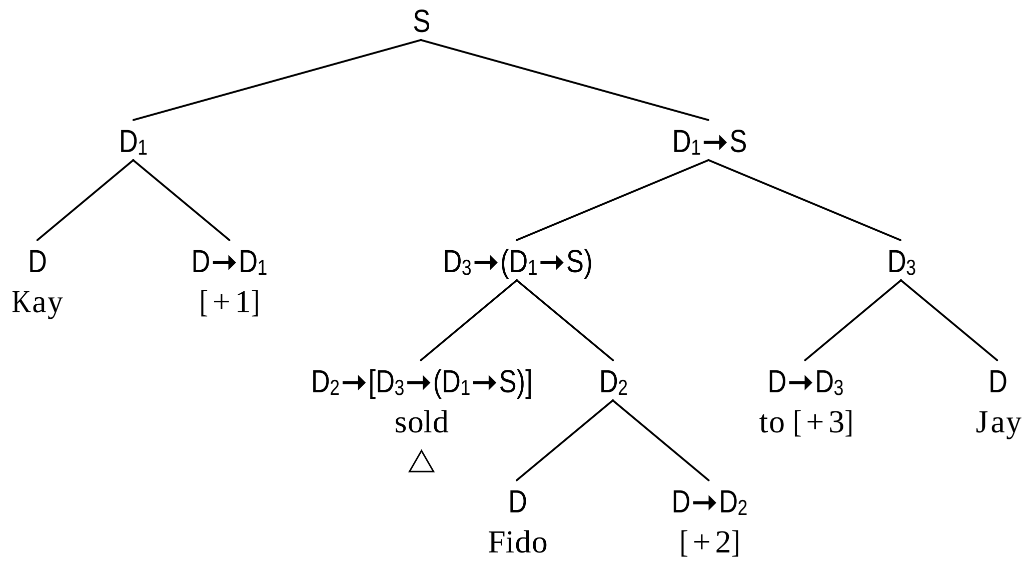
1. Jay respects Kay



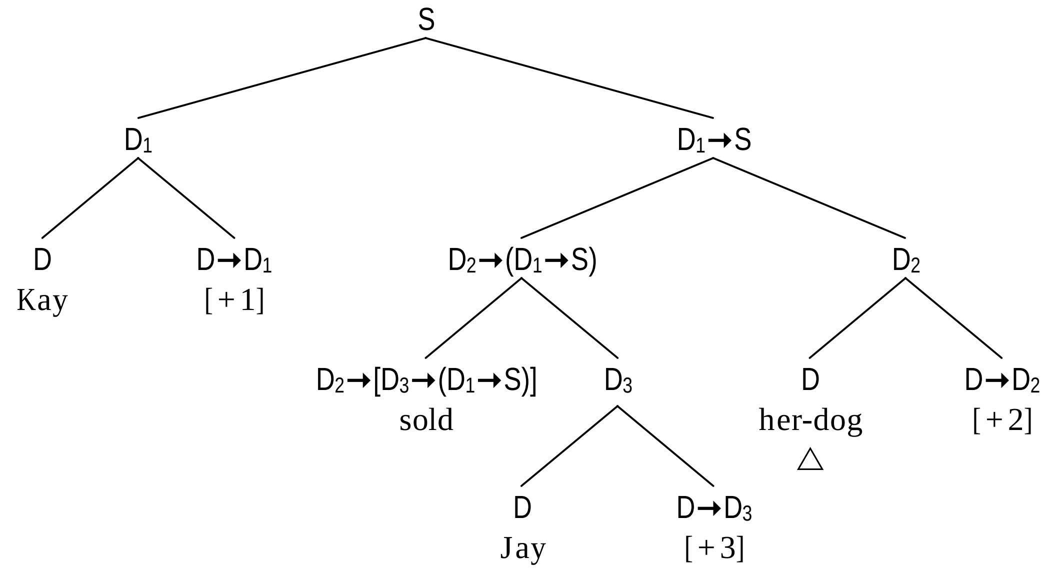
2. Jay bought Fido from Kay



3. Kay sold Fido to Jay

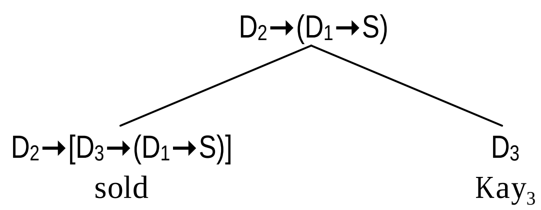


4. Kay sold Jay her-dog



2. Type Mismatch!

We now face an immediate and serious categorial problem. In particular, in the previous example, notice the following sub-tree.

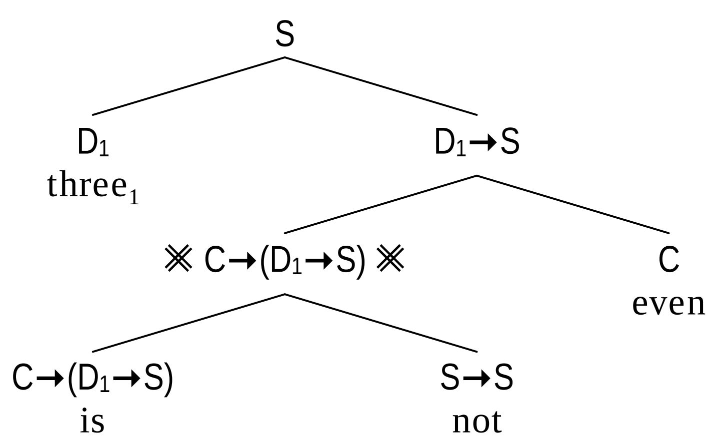


Notice in particular that ‘sell’ officially takes an accusative argument, but ‘Kay$_3$’ is dative. Thus, we have a type mismatch.

3. Other Examples of Type-Mismatch

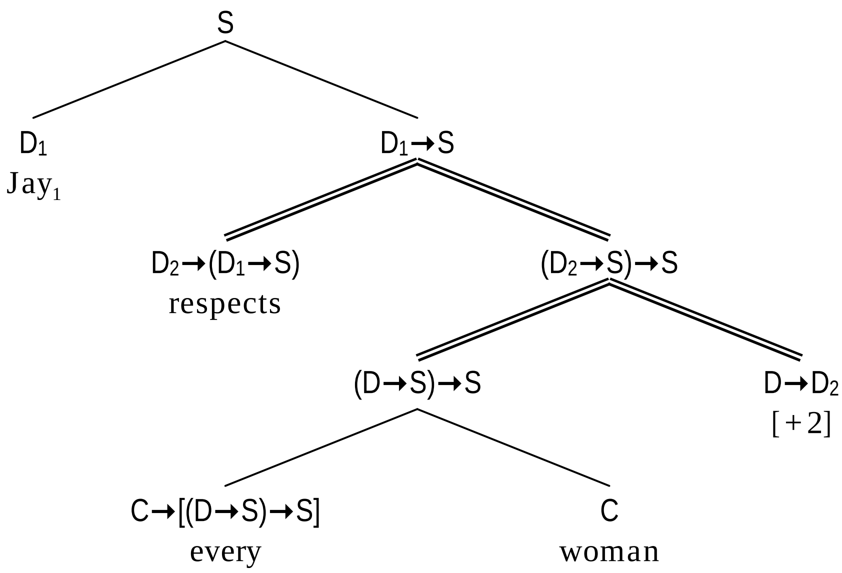
1. Misplaced Negation

three is not even



2. Accusative QPs

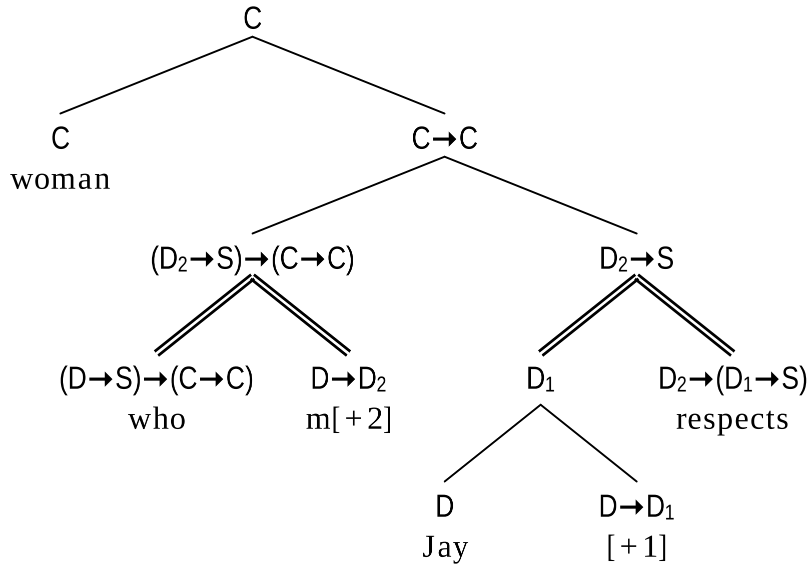
Jay respects every woman



The double-line branches indicate that the composition does not follow the current rules for composition; in particular, neither phrase serves as an argument for the other phrase.

3. Accusative Relative Pronouns

woman **whom** Jay respects



4. Revision of Composition Principles

We propose to solve this problem by expanding grammatical-composition.

1. The Fundamental Principle of Standard Categorial Grammar

a phrase of type	$\mathcal{A} \rightarrow C$
combines with a phrase of type	$\mathcal{A}$
to form a phrase of type	C
No other combinations are allowed.	

If we think of the arrow as an if-then connective, then this mode of composition corresponds to the logical inference principle known as *modus ponens*.

2. The Fundamental Principle of Expanded Categorial Grammar

a phrase of type	$\mathfrak{I}_1$
and a phrase of type	$\mathfrak{I}_2$
combine to form a phrase of type	$\mathfrak{I}_0$
if and only if	
the argument $\ulcorner \mathfrak{I}_1 \,;\, \mathfrak{I}_2 \,/\, \mathfrak{I}_0 \urcorner$ is valid according to **Categorial Logic**	

More generally:

phrases of types	$\mathfrak{I}_1, \ldots, \mathfrak{I}_k$
combine to form a phrase of type	$\mathfrak{I}_0$
if and only if	
the argument $\ulcorner \mathfrak{I}_1 \,;\, \ldots \,;\, \mathfrak{I}_k \,/\, \mathfrak{I}_0 \urcorner$ is valid according to **Categorial Logic**	

where

Categorial Logic
$=_{df}$
the logical system that **properly** models grammatical-composition (whatever that happens to be)

5. Initial Candidates

Some well-known initial candidates include:

(1)	classical logic	CL	Frege, Russell, …
(2)	intuitionistic logic	IL	Brouwer
(3)	relevance logic	RL	Anderson & Belnap
(4)	linear logic	LL	Jean-Yves Girard
(5)	strict-entailment logic	SEL	C.I. Lewis
(6)	subjunctive-conditional logic	SCL	D. Lewis, Stalnaker

6. CL, IL, and SCL do not Properly Model Grammatical-Composition

The following is valid according to CL, IL, and SCL (but not SEL, RL, or LL).[1]

$$P \,;\, Q \vdash P{\rightarrow}Q$$

If this inference-form models grammatical composition, then we have the following principle.

a phrase of type	D
and a phrase of type	S
combine to form a phrase of type	D→S

It seems highly implausible that a one-place predicate results when a name is combined with a sentence. Thus, CL, IL, and SCL do not properly model grammatical-composition.

7. No Monotonic Logic Properly Models Grammatical-Composition

Monotonicity	Adding premises to a valid argument never results in an invalid argument.
Special Case	$\mathcal{B} \vdash \mathcal{C} \;\Rightarrow\; \mathcal{A} \,;\, \mathcal{B} \vdash \mathcal{C}$
Identity	$\mathcal{B} \vdash \mathcal{B}$
SC+Id	$\mathcal{A} \,;\, \mathcal{B} \vdash \mathcal{B}$
Instance	S ; D ⊢ D

If this inference form models grammatical composition, then we have the following principle.

a phrase of type	S
and a phrase of type	D
combine to form a phrase of type	D

The implausibility of this leads us to reject any monotonic logic as a model of grammatical-composition.

[1] The symbol '⊢' – called "turnstile" – denotes the **logical consequence relation**. For example, ⌜P ; Q ⊢ P→Q⌝ means that the formula P→Q is a logical consequence of the formulas P and Q, [alternatively, the argument ⌜P ; Q ; therefore, P→Q⌝ is valid] according to the tacitly understood logical system.

8. Examples of Non-Monotonic Logics

1. Relevance Logic

The most prominent example of a non-monotonic logic is Relevance Logic (RL), which adopts the following desiderata for reasoning.

(1) In arriving at consequent $\mathbb{C}$ from antecedent $\mathbb{A}$, one must *use* $\mathbb{A}$;
(2) In arriving at conclusion $\mathbb{C}$ from premises $\mathbb{P}_1, \ldots, \mathbb{P}_k$, one must *use* every premise **at least once**.

Too Strong:

$A\to(A\to B)\ ;\ A \vdash B$

2. Linear Logic

(2*) In arriving at conclusion $\mathbb{C}$ from premises $\mathbb{P}_1, \ldots, \mathbb{P}_k$, one must *use* every premise **exactly once**.

Too Strong:

$(A\to A)\to B \vdash B$

3. Hyper-Relevance Logic

(2**) In arriving at conclusion $\mathbb{C}$ from premises $\mathbb{P}_1, \ldots, \mathbb{P}_k$, one must *use* every premise **exactly once**, and some premise(s) must be used [$k\neq0$]

Too Weak – the following desirable composition is not valid in LL or HL.

a phrase of type	$S\to(S\to S)$
and two phrases of type	$D_1\to S$
combine to form a phrase of type	$D_1\to S$

Example:

and	$S\to(S\to S)$
is virtuous	$D_1\to S$
is happy	$D_1\to S$
is virtuous and is happy	$D_1\to S$

9. Summary of Logical Systems Considered So Far

Note: in the following we consider an additional connective $\times$, which corresponds to conjunction in IL and CL, and corresponds to "fusion" in RL and below. The Key principle concerning $\times$ is the residual law, which we propose to call 'Schönfinkel's Law'.

1. Arguments Valid in all Systems

(1) $B \to C\ ;\ A \to B \vdash A \to C$ [Transitivity]
(2) $A \to (B \to C) \vdash B \to (A \to C)$ [Permutation]
(3) $A \to (B \to C)\ ;\ B \vdash A \to C$ [Secondary *Modus Ponens*]
(4) $A \vdash (A \to B) \to B$ [Lifting]
(5) $(A \to C) \to D\ ;\ A \to B \vdash (B \to C) \to D$ [Inflection]
(6) $(A \to C) \to D\ ;\ A \to (B \to C) \vdash B \to D$ [Permutation + Transitivity]
(7) $B \to C\ ;\ A \times B \vdash A \times C$ [Addition]
(8) $(A \times B) \to C \dashv\vdash A \to (B \to C)$ [Schönfinkel's Law]

2. CGL-Valid Arguments Rejected by LL

(1) $A \to (B \to C)\ ;\ A \to B \vdash A \to C$ [Conditional Modus Ponens]
(2) $A \to B\ ;\ A \to C \vdash A \to (B \times C)$ [Conditional Multiplication]
(3) $A \to B\ ;\ A \to C\ ;\ (B \times C) \to D \vdash A \to D$ [Generalized Conjunction]

3. LL-Valid Arguments Rejected by CGL

(1) $(A \to A) \to B \vdash B$ [Law of Assertion]

4. RL-Valid Arguments Rejected by LL

(1) $A \to (A \to B) \vdash A \to B$ [Contraction]
(2) $A \vdash A \times A$ [Duplication]

5. IL-Valid Arguments Rejected by RL

(1) $A \vdash B \to A$ [Positive Paradox]
(2) $A \to B \vdash A \to (A \to B)$ [Expansion]
(3) $A \vdash B \to B$ [Tautology]
(4) $A \times B \vdash A\ ;\ \ A \times B \vdash B$ [Simplification]

6. CL-Valid Arguments Rejected by IL

(1) $(A \to B) \to B \vdash (B \to A) \to A$ [Lukasiewicz's Law]
(2) $(A \to B) \to A \vdash A$ [Peirce's Law]