Top-down derivation, recursion, and the model of grammar

Jan-Wouter Zwart
University of Groningen

Complex Sentences, Types of Embedding, and Recursivity
Konstanz, March 6 2012

1. Introduction

(1) Douglas Hofstadter (2007:83) on the evolution of human cognition:

"**Concepts** in the brains of humans acquired the property that they could get **rolled together** with other concepts **into larger packets**, and any such larger packet could then **become a new concept** in its own right. In other words, **concepts could nest inside each other hierarchically**, and such nesting **could go on** to arbitrary degrees."

(2) key > keyboard > computer > desk > office > building > university > higher education > etc

(3) a. entities can be simplex and complex at the same time
b. we can jump back and forth between the simplex/complex interpretation, depending on the cognitive task at hand

(4) constituency: [the man] has that simplex/complex ambiguity
simplex: constituency tests
complex: subconstituents (created by Merge)
selective treatment, e.g. casemarking *de**r** Mann*

(5) same is true of *compounds*, *complex words*, *idiomatic phrases*, *clauses*, etc.

(6) **Question**
For any grammatical operation ***M*** (like *Merge*), manipulating a syntactic object ***SO***, consisting of subparts ***p^1, ..., p^n***, we have to decide whether M manipulates SO or p^x

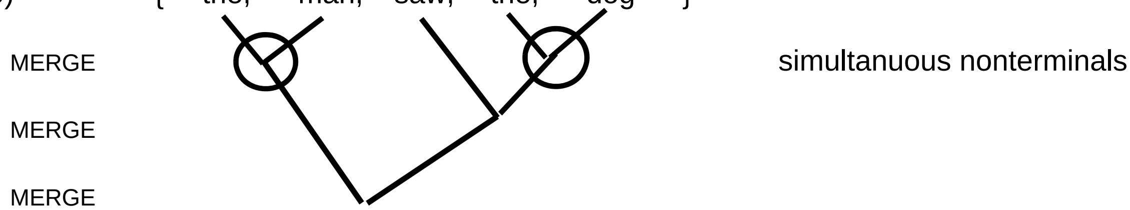
(7) The man saw the dog

potential numerations
a. { the, man, saw, the, dog }
b. { [the man], saw, the, dog }
c. { the, man, saw, [the dog] }
d. { [the man], saw, [the dog] }
e. { the, man, [saw the dog] }
f. { the, [man saw], the dog } etc.

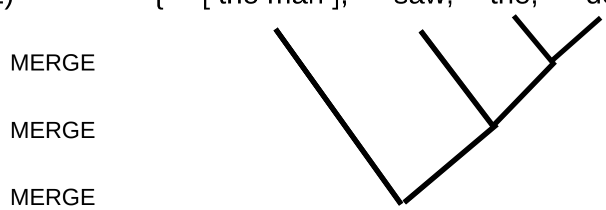
(8) We know that the structure is [[the man] [saw [the dog]]]

(9) to get to (8) starting from (7a), we need a **context-free grammar**
to get to (8) starting from (7b), a **finite state grammar** suffices
(given some minimalist conception of phrase structure rules, i.e. Merge)

(10) { the, man, saw, the, dog }



(11) { [the man], saw, the, dog }

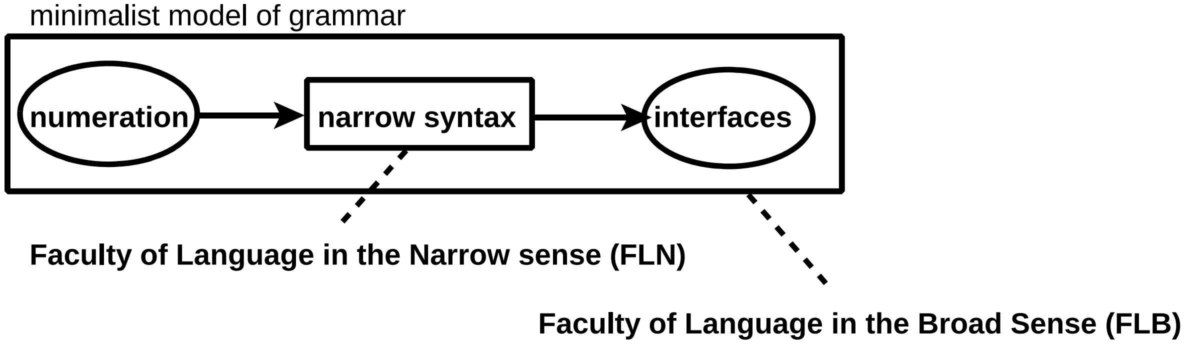


(12) crucial point: the fact that *the man* is complex does not make it a nonterminal

2. Arguments against finite state grammar

(13) Chomsky (1957:21): "English is not a finite state language."

(14) minimalist model of grammar



(15) new question: are the rules of FLN (=Merge) of the finite state type ?

(16) Chomsky hierarchy of languages/grammars (Chomsky 1956)

a.	type 3	finite state grammar	A → a | a B	(X = nonterminal, x = terminal)
b.	type 2	context-free grammar	A → (a)*(B)*	(any number of (non)terminals)
c.	type 1	context-sensitive grammar	same as b. with context added	

(17) The argument for concluding that type 3 does not suffice (Chomsky 1956)

(18) if it rains then it pours — a⁀b
if if it rains then it pours then it pours — a⁀a⁀b⁀b
if if if it rains then it pours then it pours then it pours — a⁀a⁀a⁀b⁀b⁀b
(if)^m it rains (then it pours)^n | m = n — (a)^m⁀(b)^n | m = n

(19) The interdependency of the expansion of *a* and *b* cannot be expressed in a finite state grammar

(20) The argument for concluding that type 2 does not suffice (Huybregts 1976)

(21)

We	hebben	de kinderen	Hans	het huis	laten	helpen	verven
we	have	the kids	Hans	the house	CAUS	help	paint

‘We let the children help Hans to paint the house.’ (Dutch)

(22) A context-free grammar can generate strings with the right number of NPs and verbs, but cannot ensure that the NPs and verbs are lined up in the correct order.

(23) These arguments presuppose:
a. the basic building blocks of syntax (the alphabet/numeration) are **words**
b. a sentence is derived in a single run

(24) This paper: if you give up these assumptions, the arguments disappear

3. Merge

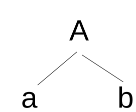
(25) What is it that syntax must derive?
a. constituency
b. relations (configurationally determined dependency)
c. hierarchy

(26) DASR = Derivational Approach to Syntactic Relations (Epstein 1999):
Merge can do all this

(27) common conception of merge

(i) Numeration N = { a, b, c, d, e }

(ii) Merge (a) first merge: take 2 elements from N and combine them
(b) other merge: take 1 element from N and combine it with A



(28) arbitrary elements:
(iia) why 2 elements ? (only way to derive constituency)
(iib) why with A ? (only way to prevent wayward derivations)

(29) removing the arbitrary elements of (28) (Zwart 2004):

Merge: move 1 element at a time from the Numeration to the Workspace (=A)

STEP	NUMERATION	WORKSPACE
1.	{ a, b, c, d, e }	--
2.	{ b, c, d, e }	a
3.	{ c, d, e }	⟨b, a⟩
4.	{ d, e }	⟨c, ⟨b, a⟩⟩
etc.		

(30) Gives you:
a. constituency (a constituent is a stage of the workspace)
b. dependency (merge is asymmetric, yields an ordered pair)
c. hierarchy (a function of the relative timing of merger of elements from N)

(31) Keeps one arbitrary element, implicit in the standard conception of merge:
transfer between Numeration and Workspace

(32) Bobaljik (1995): merge does not do any transfer, it merely specifies relations among the members of the Numeration

STEP	MERGE	NUMERATION
1.		{ a, b, c, d, e }
2.	a + b	{ a, b, c, d, e, a+b }
3.	c + d	{ a, b, c, d, e, a+b, c+d }
4.	e + (a+b)	{ a, b, c, d, e, a+b, c+d, e+(a+b) }
etc.		

(33) enriching the numeration > increases the number of combinatorial possibilities
(also keeps some of the arbitrary elements of standard merge)

(34) top-down derivation (**split merge**)
(i) no transfer (affecting only the numeration)
(ii) manipulates one element at each step
(iii) reduces possibilities at each step (finite and directed process)

(35) Split Merge

STEP	SPLIT	NUMERATION	ORDERED PAIR/DEPENDENCY
1.	--	{ a, b, c, d, e }	--
2.	a	{ b, c, d, e }	〈 a, { b, c, d, e } 〉
3.	b	{ c, d, e }	〈 a, 〈 b, { c, d, e } 〉〉
4.	c	{ d, e }	〈 a, 〈 b, 〈 c, { d, e } 〉〉〉
etc.			

(36) This yields:
a. constituency (a constituent is a stage of the numeration)
b. dependency (the ordered pair after split merge)
c. hierarchy (achieved by the relative timing of split)

(37) Potentially very big problem: no natural place for movement in this system

4. The Numeration

(38) The Numeration is not a set of words

(39) { the, man, saw, the, dog } (= (7a))

split 1: 〈 the, { man, saw, the, dog } 〉 *man saw the dog* is not a constituent

so the numeration must be (7b) = { [the man], saw, the, dog } (or some variant)

(40) Observations showing that the numeration need not be homogeneous (cf. Ackema and Neeleman 2004)

a. word + morpheme **vader** ‘father’ + **-je** DIM > **vader-tje** ‘dear/little father’ (Dutch)

b. phrase + morpheme **vader en moeder** ‘father and mother’ + **je** DIM > [**vader-en-moeder**]-**tje** ‘playing house’ (Dutch)

c. phrase + word **Sturm und Drang** ‘Storm and Stress’ + **Gefühl** ‘feeling’ > **[Sturm-und-Drang]-gefühl** ‘Storm and Stress-feeling’ (German)

d. clause + word **ik weet niet wat ik moet doen** ‘I don’t know what to do’ + **gevoel** ‘feeling’ > [**ik weet niet wat ik moet doen**]-**gevoel** ‘feeling of not knowing what to do’ (Dutch)

e. clause + morpheme **ik weet niet wat ik moet doen** ‘I don’t know what to do’ + **ge** FREQ/ITER > **ge**-[**ik weet niet wat ik moet doen**] ‘constantly letting on that you don’t know what to do’ (Dutch)

(41) *Uniformity hypothesis*
Every structured complex is created by Merge

> input to Merge is nonhomogeneous

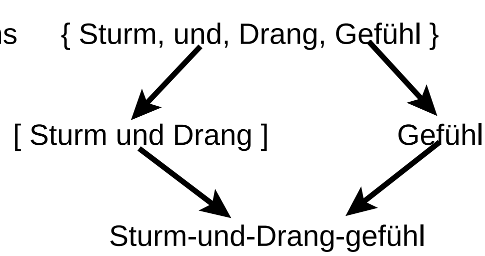
(42) Approaches to nonhomogeneity

	# OF NUMERATIONS	# OF DERIVATIONS	TYPE OF OPERATION
internal	1	1	enriching the numeration (cf. (32))
parallel	1	> 1	multiple workspaces (using transfer)
serial loop	> 1	> 1	output becomes input (recursion)

(43) Difference: only the serial loop incorporates interface processes, i.e. the creation of (potentially idiosyncratic) sound-meaning pairings

(44) a. [Sturm-und-Drang]-gefühl (= (40c))

b. parallel derivations



> but *Sturm und Drang* has an idiomatic meaning, which must be established at the interfaces (i.e. **between** derivations)

c.

NUMERATION 1	{ Sturm, und, Drang }
DERIVATION 1	〈 Sturm, 〈 und, 〈 Drang 〉〉〉
INTERFACE 1	[Sturm und Drang] = pre-Romantic movement

↓

NUMERATION 2	{ [Sturm und Drang], Gefühl }
DERIVATION 2	〈 [Sturm und Drang], Gefühl 〉
INTERFACE 2	[Sturm und Drang gefühl]

(45) *Sturm-und-Drang* has the simplex/complex ambiguity referred to at the beginning

(46) 2 arguments for layered derivations:
a. complex left branch elements (*the man saw the dog*) [given a simple form of merge]
> includes subjects, adjuncts
> arguably extends to conjuncts (complex entities treated as single items)

b. complex elements with idiosyncratic sound/meaning properties
> includes 'constructions'

5. Recursion

(47) **recursion** on this approach
a. merge (split-merge, (35)) is not recursive but iterative

> in fact, of the finite-state type A → a B

b. serial loop (derivation layering, (44c)) is prototypical recursion

> an item in a numeration for a derivation encapsulates an entire derivation

6. Revisiting the argument against the grammar being finite state (FS)

(48) a. if it rains then it pours
b. if if it rains then it pours then it pours
etc.

(49) Assume that this is essentially coordination:

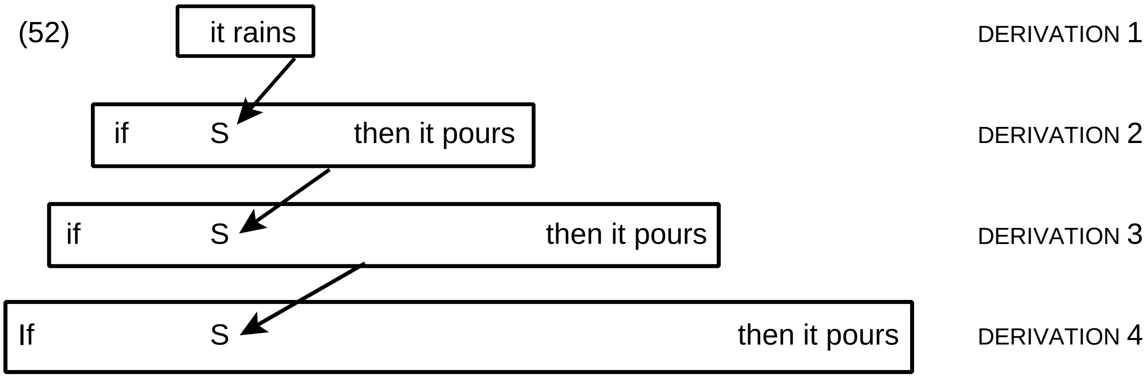
[if it rains] [then it pours] > NUMERATION { [if it rains], [then it pours] }

SPLIT MERGE 〈 [if it rains], then it pours 〉 = FS

(50) [if it rains] = output of separate derivation, where *it rains* = S

(51) S may itself be a conditional coordination like (49), the *if*-clause of which contains S, etc.

(52)



(53) the equal number of *if*-clauses and *then*-clauses follows from the recursive process:

> you stick and *a*-*b* pair inside the *a* of another *a*-*b* pair every time

(54) recursion (in terms of derivation layering) gives you a network of FS-grammars with the strong generative capacity of a higher order grammar

7. Addressing the ± context-free discussion

(21)

We	hebben	de kinderen	Hans	het huis	laten	helpen	verven
we	have	the kids	Hans	the house	CAUS	help	paint

‘We let the children help Hans to paint the house.’ (Dutch)

(55) *simpler version*

We	hebben	de kinderen	Hans	laten	helpen
we	have	the kids	Hans	CAUS	help

‘We let the children help Hans.’

(56) *constituency?*

a.	[laten helpen]	hebben	we	de kinderen	Hans	(niet)
	let help	have	we	the kids	Hans	not
b. *	[Hans helpen]	hebben	we	de kinderen	(niet)	laten
	Hans help	have	we	the kids	not	let

(57) the numeration must include the verb cluster as a single item

(58) { we, [de kinderen], Hans, [het huis], [hebben laten helpen verven] }

NB, this assumes that verb second is an interface effect (Chomsky 2001, Zwart 2005)

(59) (21) can be generated via split merge (=FS)

〈 we, 〈 [de kinderen], 〈 Hans, 〈 [het huis], 〈 [hebben laten helpen verven] 〉〉〉〉

(60) word order:
a. order of NPs is fixed = grammatical function hierarchy
b. order of verbs is variable (across dialects) = spell-out effect

> the true cases of cross-serial dependencies are accidental

8. The model of grammar

(61) A grammar includes at least the following processes:

a. numeration composition
b. split merge
c. interface processes (linearization, morphological marking, reanalysis, atomization)

(62) Of these, split merge (61b) can be described as a finite state grammar

{ a, b, c, d, e } = nonterminal A

a { b, c, d, e } = terminal⌒nonterminal a B

(63) The whole package (61a-c) certainly is more complicated, but not of the type that can be located on the Chomsky hierarchy of grammars/languages

(64) Merge in its simplest form is not recursive but iterative

> is there evidence that it should be of a higher complexity?

(65) If recursion (applied to communication) is what defines the human language faculty (Hauser/Chomsky/Fitch 2002), the key element is not merge (narrow syntax) but the ability to connect derivation layers (recursive loop).

(66) This is the ability to treat a complex structure as a single item, and to move back and forth between the complex and atomic interpretation, depending on the cognitive task at hand.

References

Ackema, Peter and Ad Neeleman. 2004. *Beyond morphology*. Oxford University Press.
Chomsky, Noam. 1956. Three models for the description of language. *IRE Transactions on Information Theory* Vol. IT-2.
Chomsky, Noam. 1957. *Syntactic structures*. Mouton.
Chomsky, Noam. 2001. Derivation by phase. In Mike Kenstowicz, ed., *Ken Hale: a life in language*. MIT Press.
Epstein, Samuel D. 1999. Un-principled syntax and the derivation of syntactic relations. In Samuel D. Epstein and Norbert Hornstein, eds., *Working minimalism*. MIT Press.
Hauser, Marc D., Noam Chomsky, and W. Tecumseh Fitch. The Faculty of language: what is it, who has it, and how did it evolve? *Science* 298, 1569-1579.
Hofstadter, Douglas. 2007. *I am a strange loop*. Basic Books.
Huybregts, Riny. 1976. Overlapping dependencies in Dutch. *Utrecht Working Papers in Linguistics* 1, 23-65.
Zwart, Jan-Wouter. 2004. The format of dependency relations. Bloomington lectures, June-July.
Zwart, Jan-Wouter. 2005. Verb second as a function of merge. In Marcel den Dikken and Christina M. Tortora, eds., *The function of function words and functional categories*. Benjamins.

Center for Language and Cognition Groningen
Faculty of Arts, PO Box 716, NL9700AS, Groningen, The Netherlands
www.let.rug.nl/zwart ● user: c.j.w.zwart, at: rug.nl