Croatian Journal of Philosophy
Vol. XXIV, No. 71, 2024
https://doi.org/10.52685/cjp.24.71.5
Received: March 19, 2024
Accepted: May 13, 2024

Generative Linguistics and the Computational Level

FINTAN MALLORY
Durham University, Durham, UK

Generative linguistics is widely claimed to produce theories at the level of computation in the sense outlined by David Marr. Marr even used generative grammar as an example of a computational level theory. At this level, a theory specifies a function for mapping one kind of information into another. How this function is computed is then specified at the algorithmic level before an account of how this is algorithm is realised by some physical system is presented at the implementation level. This paper will argue that generative linguistics does not fit anywhere within this framework. We will then look at several ways researchers have attempted to modify either the framework of generative theory to reconcile the two approaches. Finally, it presents and discusses an alternative position, anti-realism about generative grammar. While this position has attracted some recent support, it also runs into some of the problems that earlier modifications faced.

Keywords: Generative grammar; Marr; computation; cognitive science; linguistics; Chomsky.

What is the relation between generative linguistics and the rest of the cognitive sciences? Despite the historical role played by generative grammar in the cognitive turn of the 1950s and 60s, linguists have complained for decades that the rest of the cognitive sciences largely ignore their findings.[1] This concern has only intensified with the revival of connectionism in cognitive science under the guise of "artificial intelligence" (i.e. deep neural networks). Consider the recent claim that

[1] For example, Ian Roberts asks "why is mainstream generative syntax overlooked in cognitive science as a whole?" (Roberts 2014: 22) or as Ray Jackendoff's phrased it in the title of a Topic-Comment piece from 1988, "Why are They Saying These Things about Us?" (Jackendoff 1988).

"After decades of privilege and prominence in linguistics, Noam Chomsky's approach to the science of language is experiencing a remarkable downfall" (Piantadosi 2023). Behind these concerns is a lack of clarity about the metatheory of generative linguistics, i.e. claims about what generative models—grammars—actually model, and specifically, in what sense they describe computations. For decades it has been customary to claim that generative grammar was a computational level theory akin to David Marr's program within the cognitive neuroscience of vision. If so, generative grammarians would be seeking the same kinds of explanations that have had success across the cognitive sciences.

A glance at the literature would suggest that this is the case. Marr explicitly invoked generative grammar as an example of a computational level theory when he introduced his levels of analysis writing that "Chomsky's (1965) theory of transformational grammar is a true computational theory in the sense defined earlier" (Marr 1982: 28). Chomsky, in turn, agreed claiming, "We may consider the study of grammar and UG [universal grammar] to be at the level of the theory of computation" (Chomsky 1982: 48). This idea remains the consensus position among linguists. As Klaus Abels puts it "theorising at the most abstract, the computational-level has remained the mainstay of work in theoretical linguistics."[2] In the last few years, appeal to Marr has been used to justify or explain theoretical disagreements between traditional generative grammarians and usage-based theorists (Yang 2017, Adger and Svenonius 2015) as well as model-theoretic syntax (Neeleman 2013; Graf 2017). It has appeared in debates concerning how the levels of description should interact in linguistics (Yang 2017; Hornstein 2013; Abels 2013; Hornstein and Pietroski 2009), has been invoked in debates about language evolution (Johnson 2015, 2016; Berwick and Chomsky 2016; Perfors 2017), as well as the independence of knowledge from production and comprehension (Neeleman and Van de Koot 2010). Appeal to Marr has also formed the basis of the interaction between theoretical linguistics and other subfields within cognitive science (Poeppel 2017; Kobele 2012; Embick and Poeppel 2014; Jackendoff 2012; Murphy 2015). The idea that generative grammar is a computational level theory in Marr's sense constitutes the most successful response to the "realism" debates which have followed generative linguistics since its inception (see Pylyshyn 1973 for a discussion of the "psychological reality" of generativist claims).

This paper will first argue that, despite widespread claims to the contrary, generative grammar is not a computational-level theory in

[2] See also, "What Chomsky (1965) calls a theory of 'competence' or 'knowledge of language' corresponds to Marr's computational theory" (Jackendoff 2012: 1133). "[t]he competence theories of linguistics correspond to Marr's (1980) topmost level of computational theory [...]" (Heinz 2011: 140). "A theory of grammar corresponds to Marr's abstract theory of a computation" (Berwick 1985: 9). Countless other examples of this claim can be found in the literature.

the sense articulated by Marr (sections 1-4). Simply put, Marr's computational level concerns a theory of performance, not competence whereas generative grammar purports to describe linguistic competence. I will then go on to consider the alternative interpretations of generative grammar that have been proposed; that generative grammar is a metamathematical theory of computation, a theory of parsing (i.e. performance), and a description of a separate data structure utilised by the parser. While each of these approaches has its merits, we will see that each requires rejecting core features of generative linguistics. Finally, I will articulate a position that is, at least, implicit in some core generative literature, that of *modal anti-realism* about computation. This position makes sense of some theoretical practice but ultimately denies that the structure-building operations posited by grammarians are realised as processes in the human brain.

2. *Marr's computational level*

According to David Marr, information processing systems are best described at three different levels: computational, algorithmic, and implementation. At the computational level, "the performance of the device is characterised as a mapping from one kind of information to another, the abstract properties of this mapping are defined precisely, and its appropriateness and adequacy for the task at hand are demonstrated" (Marr 1982: 24). The mapping is stated as a function: $f: I \rightarrow O$ (hence the alternative name "function-theoretic explanation"). At the algorithmic level, a representation of the input and output of this mapping is provided, and an algorithm that produces the output from the input is proposed. At the implementation level, an account is given of how this algorithmic process is physically realised by some physical system, e.g. the activations of neurones, oscillatory dynamics, transistors, etc. While these levels are conceptually distinct, the development of a theory at one level may inform theory construction at others (for a brilliant demonstration of how this works, see Jonas and Kording (2017).

Standard examples of computational-level theories are the analysis of the auditory system in terms of Fourier transforms (e.g. Schneider and Mores 2013), hand-eye coordination using vector subtraction (Perrone and Krauzlis 2008), Marr's model of edge detection, and the use of path integration by various animals (Eteinne and Jeffery 2004). So that we have a concrete example to work with, we'll consider Marr's own example of a computational level analysis of a cash register. While this is rough and simple, it shall serve our purposes going forward.

Imagine we want to understand a cash register. At the computational level, we might note that a cash register computes addition. This computational-level characterisation is blind to questions of representation, e.g. whether the cash register uses binary or Roman or Arabic numeral systems. At the algorithmic level, a particular algorithm that computes this function is posited. The algorithm makes claims about the representational format of the information being processed. At the implementational level, it is explained how this algorithm may be realised by circuitry and micro-transistors. Returning to the computational level, the function characterised is:

(1) Addition: $f(x, y) = x + y$

The computational theory does not only specify which function is implemented by the system but also justifies *why* the function is the appropriate one for our theory. What is at issue here is not *why the system* implements the function but *why our theory* says that this function is the right one (the focus is not teleological but methodological). It is often the case that several different functions could produce the mapping identified and a why-story connects these mathematical properties with relevant features of the world or the task under consideration.[3] In the cash register example, Marr connects the algebraic properties of addition with commercial practices. There is a zero element because buying nothing costs the same as not buying anything. The order in which goods are purchased shouldn't affect the total price (commutativity), nor does it matter if they are paid for separately (associativity) and the register can also handle the existence of a refund policy (inverses).[4] What allows us to speak of the different properties of these functions is the intensionality of our characterisations, i.e. the function is described independently of its inputs and outputs. As Egan observes, one might use an algorithm to specify a function at the computational level without making a commitment to the algorithm which implements the function (Egan 2017). For now, we note two important things that follow from the intensionality of the computational level description.

First, having an intensional characterisation of the function allows us to discuss the function independently from the environment in which it is embedded—where "environment" may be understood as

[3] Example 1: In Marr's account of stereopsis (i.e. binocular vision), he takes into account where dots may actually appear on physical surfaces in the world in order to select between functions: "We have to examine the basis in the physical world for making a correspondence between the two images" (Marr 1980: 112). Example 2: Perrone and Krauzlis' (2008) account of eye rotation. The task modelled is the subtraction of image movement as a result of eye-rotation from image movement that occurs as a result of an agent's traversal through space, i.e. vector subtraction. One way of answering the question of how vector subtraction occurs in the brain involves the use of arctan (the inverse tangent function) while another involves treating the vectors as cosine curves in which their length and direction correspond to the amplitude and phases of the curves. The authors observe that "the problem of singularities associated with the inverse tangent also seems to preclude any simple biological implementation" and so choose the second approach.

[4] While Marr claims that these properties uniquely individuate the operation of addition, they will actually hold of any operation on an abelian group and while Marr's initial claims that the "why"-part of computational-level theorising can individuate a unique function is clearly too strong, it can be amended (see Anderson on Rational Analysis, Anderson 1990).

either the physical environment in which the agent is located or the wider properties of the information-processing system (or cognome). For example, cash registers don't compute the addition function for values less than some bound and "error" for values above it. Adding or removing memory from our cash register without altering the addition algorithm will alter the function's domain and range but intensionally it won't change the function computed. Likewise, the visual system presumably computes the same function for edge-detection whether one has glaucoma or not.

Second, the fact that the function can be characterised independently of the actual activity of the system serves a normative purpose. Once it has been determined what function the system computes, the theorist is in the position to assess if the system is functioning normally (Egan 2017). Supporting this is the fact that the mathematical theory which characterises the function is independent of the psychological theory which describes its implementation. In the example above, the theory of arithmetic is not grounded by the theory of cash registers. There is no suggestion that 1+1=2 because that's how cash registers see the world. Rather, the cash registers are designed (though they could be naturally occurring, Darwinian-evolved cash registers) to track this independent mathematical fact. The fact that the truths of mathematics are independent of the existence of cash registers doesn't entail that cognitive systems implementing functions can only be individuated by reference to extra-mental mathematical reality. It simply acknowledges that we don't expect facts about cash registers to ground facts about numbers.

These two points will be important when we consider whether or not generative grammar provides computational level theories. First, though, we must turn to generative grammar.

3. *Generative grammar*

Generative grammar is the branch of cognitive science aimed at characterising the state of the human mind corresponding to an individual's knowledge of a language. A generative grammar is a function-theoretic characterisation of this knowledge. A quick glance at the literature would suggest that the function corresponding to a grammar characterises a mapping from sounds (or visual signs) to meanings.[5] This suggests a function along the lines of:

[5] This function is often spoken of in Marrian terms: "Generative accounts of linguistic phenomena are couched at a level of analysis that is close to Marr's (1982) computational-level. That is, the theory specifies a system that guarantees a particular pairing of sounds and meanings across a potentially unbounded domain" (Adger and Svenonius 2015: 6). "A computational account of language has two parts. First is a specification of which sounds (or more generally signals) convey which meanings" (Kobele 2012: 411).

(2) Grammar: f(sound) = meaning

However, matters are not so simple. Standard function-theoretic characterisations are methodologically possible because researchers have a pre-existing mathematical account of what the functions are; they are typically number-theoretic. Number theoretic functions are used because these functions are often defined over quantifiable inputs, e.g. Marr's appeal to the Laplacian of a Gaussian ($\nabla^2 G$) of the retinal array is possible because it is defined over (numerical) intensity values. In contrast, we do not have a pre-syntactic grasp of the sets of possible phonological and semantic structures between which our grammar characterises a mapping. Furthermore, the sets of sounds and meanings we are interested in are unboundedly large (we want to know how unboundedly many strings can map to unboundedly many meanings). As a result, generative linguists don't attempt to characterise this mapping directly. Instead, they describe an operation for building syntactic structures from lexical inputs. More accurately, they describe a function that recursively enumerates the set of ordered ⟨sound, meaning⟩ pairs where each element of this set is individuated by its syntactic structure. This is still a characterisation of the sound-meaning function but "from below."

The input to this function is typically presented as either a finite set of lexical items, either the whole lexicon as in Collins and Stabler (2016) or as a numeration. I'll represent it here as the power set of the lexicon (strictly speaking, this should include multisets, see Adger 2021), while the output will be the set of structural descriptions (SDs), or syntactic structures of the language.

(3) Grammar: g(lexicon) → Syntactic Structures

The grammar recursively enumerates the set of sound-meaning mappings as structured by the syntax of the language. In effect, it decomposes the function *f*, by revealing the structure of each sound-meaning pair.[6]

To know a language, according to generative linguistics is, in part, to realise a function that outputs the set of syntactic structures for that language. This function is, in turn, defined by describing an operation for combining items in the lexicon into more complex structures. This operation can apply iteratively to the structures it has already generated, as we see in the following toy example where we imagine that $\oplus$ is the structure-building operation:

[6] In other words, function *g* recursively enumerates the extension of function *f*. Instead defining a mapping from a set of sounds to a set of meanings (sounds as inputs, meanings as outputs), function g takes ordered sound-meaning pairs (i.e. lexical items) as the primitives and enumerates the set of possible combinations of sound-meaning pairs. It should be noted that this is also the case in systems like HPSG in which phonological and semantic information are combined in the same feature structure.

(4) “the” $\oplus$ “dog” $\rightarrow$ [the dog]
(5) “likes” $\oplus$ “the dog” $\rightarrow$ [likes[the dog]]
(6) “the cat”$\oplus$ “likes the dog” $\rightarrow$ [[the cat] [likes [the dog]]]

Along with a simple operation for merging syntactic objects, grammars are also capable of moving these items to different locations in the syntactic structure, thereby building more complex structures. For example:

(7) “$\underline{\text{the dog}}_1$” + “the cat likes $\underline{\text{the dog}}_1$” $\rightarrow$ [[the dog][the cat] [likes [~~the dog~~]]]

In (7), the noun phrase “the dog” which had its grammatical case determined by the verb “likes,” has been raised to form the relative clause “the dog the cat likes.” As a result of movement, even though “the dog” was the first element in the phrase to be constructed, it finds itself at the (linear) beginning of the phrase. Within minimalist syntax, it is possible for the most embedded element to be the first merged. Furthermore, which structures can be built by the structure-building operations will be determined by *syntactic features* of the lexical items they take as their input. Just as the combinatorial capacities of legos and atoms are determined by the intrinsic properties of those entities, the combinatorial properties of lexical items, and thus what the structure-building operation can do with them, are determined by their syntactic features (e.g. an item with the features V, =N is labelled as V and can combine with an item labelled as N).

Within Minimalism, the core structure-building operation is called “merge,” within Head Driven Phrase Structure Grammar and Generalised Phrase Structure Grammar, it is called “unification,” within Tree-Adjoining Grammars, it is “tree adjunction,” within categorial grammars, it is “function composition.”[7] The exact details of these frameworks aren’t relevant to this discussion; instead, what matters is that they all characterise the knowledge of language in terms of an operation for building syntactic structures. I will be using “merge” as a cover-all term for *the* core syntactic operation in what follows. By characterising a function that outputs unboundedly many different syntactic structures, the linguist, in theory, gives an account of what is involved in knowing a language. The question is whether describing

[7] Within minimalism, merge and movement are constrained by a series of further feature-checking operations (agree, probe, labelling etc.) which determine whether two lexical items can be merged, but ultimately, it is merge that builds the syntactic structures. Not every framework has “movement” as illustrated in our toy case but all describe a basic operation for constructing complex structures. While unification is also used in some varieties of Construction Grammar (e.g. Kay and Fillmore 1999) it is unclear how much of the discussion in this paper would apply to CxG theories which draw directly on usage. I suspect that CxG approaches will relate to both performance and the Marrian framework quite differently to more generative frameworks. For a recent discussion of the connections between CxG and the Predictive Processing model of cognition, see Michel (2023).

this function amounts to giving a computational-level theory for our knowledge of language.

Superficially, this appears to be the case. Just as with computational level theories, generative grammars provide intensional characterisations of a function: "I-languages are functions regarded in intension" (Chomsky 1995: 26). Grammars describe linguistic competence independently of the social and cognitive environments in which they are embedded including any constraints on memory facing parsers.[8] The theories are computational, in that, they specify a general method by which an output can be generated in a finite number of steps. Furthermore, generative linguistics is often involved in comparing extensionally equivalent systems of grammar and presenting arguments for why one is superior to another. For example, while multiple context-free grammars and Minimalist grammars can generate the same sets of syntactic structures, linguists have given compelling reasons to believe that the latter is the more cognitively plausible, in effect, presenting the *why* component of a computational level theory. The problem is that a grammar is not a computational level explanation.

4. *Differences between generative grammar and computational level explanations*

We'll consider here two differences between the function characterised in (3) and Marrian computational level theories.

1. Performance/Competence: The first and most obvious reason that a generative grammar is not a computation-level theory in Marr's sense is that it does not describe a process and so, by extension, does not characterise information processing. A syntactic derivation is not a real-time event. Actual linguistic processing, or at least our parsing models of it, must incrementally construct syntactic representations from unlabelled inputs, starting with the words at the start of the sentence. In contrast, the syntactic derivations posited in generative grammar build structures from the most embedded constituents outwards, applying movement operations when necessary.

Furthermore, the inputs to syntactic derivations are not unlabelled strings as the inputs to parsing are but highly specified feature structures that represent information about the elements combined (this

[8] It is worth noting that, if we embrace the Marrian interpretation of generative grammar, then the Minimalist Program would appear to be a perfectly reasonable application of Anderson's "Principle of Rationality" for computational-level theorising. This is the idea that we take the function computed by some cognitive system to be the optimal function for the task and incrementally and iteratively modify our proposal on the basis of how the system's behaviour diverges from the function proposed. This has proven to be methodologically well-motivated for narrowing down which functions should be posited at the computational level (for a recent defence of the method see Van Rooij et. al. 2018). It is also worth noting that Anderson is one of the few theorists who doesn't equate competence theories with computational-level theories (Anderson 1990: 8–9).

was left out of the example above). This disconnect between the two approaches is quite explicit. While on the computational level, "[t]he performance of the device is characterised as a mapping from one kind of information to another" (Marr 1982: 24), "[a] generative grammar [...] in no sense purports to be a description of his actual performance, either as a speaker or as a listener" (Chomsky 1965: 3). It is instead a theory of competence; an account of what an agent must *know* (or "cognise") in order to perform some task, not a computational characterisation of the task itself. It is presumably this issue that Chomsky is raising when he writes: "David Marr's influential ideas about levels of analysis do not apply here at all, contrary to much discussion, because he too is considering input-output systems [...]" (Chomsky 1995: 12).

2. Grounding: A second difference is that, unlike the mathematical functions typical of computational-level theorising, generative grammars ground the structures they output. According to standard generative assumptions, a sentence has the syntactic structure linguists ascribe to it because that is the structure assigned to it by a grammar.[9] A cognitively realised grammar makes it the case that a sentence has its particular syntactic structure and not some other one, and the structures which a grammar generates are characterised solely with reference to that grammar; there is no independent method for characterising the grammatical structures of language (at least over an infinite set). Contrast this with the cash register example above, it is clear that one needn't be a Platonist to think that the values the cash register computes don't depend on the cash register for their existence. 1 + 1 does not equal 2 because that is how cash registers see the world. Rather it's the fact that 1 + 1 = 2 is true independently of the cash register that allows us to determine whether or not the cash register is functioning properly since we can contrast the results of addition with the results of a broken cash register. The natural numbers have their structure independent of cash registers. In contrast, the syntactic structures characterised in generative grammar are function-dependent. We cannot characterise the full set of syntactic structures output by a grammar except with reference to the grammar itself.

Since syntactic structures can only be ascribed to a sentence with reference to a particular grammar, it isn't possible to distinguish between the function a grammar is supposed to compute and the function that it actually does compute (though we can still say that a sentence is ungrammatical relative to a grammar). As a result, it simply doesn't make sense to say that the grammar is computing the *wrong* function (generative grammarians are descriptivists, not prescriptivists about grammar).[10] This problem does not arise for other cognitive systems

[9] There are some realists (or Platonists) about linguistic structure who claim it exists independently of the human mind (Devitt 2006, Katz 1981, Katz and Postal 1991) but this remains a fringe position.

[10] This point shouldn't be mistaken for the familiar bugbear of functional indeterminacy. The problem of functional indeterminacy concerns whether or not

studied within the Marrian framework. We do not assume that the world is 3D because our visual system creates a 3D representation from its input whereas we do assume that a sentence has the particular structure it possesses in virtue of speakers' particular grammars. Similarly, the retinal array exists independently of the function which uses it to construct 3D objects, whereas linguistic structure does not exist independently of a grammar.

While Chomsky (1995) appears to reject the idea that generative linguistics produces computational level theories in Marr's sense, more recently Berwick and Chomsky explicitly endorse a Marrian interpretation of generative grammar and imply that the operation merge is implemented on a lower algorithmic level (Berwick and Chomsky 2016: 132-139). However, *even more recently*, Chomsky et al., (2023) cites a 2012 interview in which he does suggest that the Marrian framework is ill-suited for understanding "internal capacities." The quote cited doesn't appear in the printed version of the interview but it is worth examining:

> As discussed in Marr (1982), complex biological systems must be understood at different levels of analysis (computational, algorithmic, implementational). Here we discuss internal language, a system of knowledge, which we understand at a computational level. Since such a system is intensional, therefore not a process, there's no algorithm. In contrast, externalization, a process of using the internal system, may find an algorithmic characterization. (Chomsky et al. 2023: 8)[11]

This is by far the most explicit statement of how Chomsky regards generative grammar to relate to the Marrian framework. I suspect

it is possible to identify a correct function in cases where the function's domain of application is infinite or even just very large. The problem here, however, has nothing to do with the size of the function's input or output. It arises as a result of the widespread commitment to the mind-dependence of linguistic structure within generative grammar. If the function implemented by the language faculty is what makes it true that a sentence has the structure it possesses, that function cannot be assessed with regards accuracy.

[11] In the same interview, Chomsky considers how one would characterise knowledge of mathematics and the case is very similar to language. "If you try to find out what that internal system is of yours, the Marr hierarchy doesn't really work very well. You can talk about the computational level—maybe the rules I have are Peano's axioms that describes a core set of basic rules of arithmetic and natural numbers, from which many useful facts about arithmetic can be deduced, or something, whatever they are—that's the computational level. In theory, though we don't know how, you can talk about the neurophysiological level, nobody knows how, but there's no real algorithmic level. Because there's no calculation of knowledge, it's just a system of knowledge. To find out the nature of the system of knowledge, there is no algorithm, because there is no process. Using the system of knowledge, that'll have a process, but that's something different" (Chomsky 2012). While Chomsky is only one generative linguist among many, his ideas have been uniquely influential in the field and if he has a non-standard interpretation of what it is involved in developing a computational level theory, it may be useful for researchers working the same tradition to be aware of this, if only to reflect on what they mean when they say generative grammar is computational.

many would agree that, if there is no algorithm computing the function, it is not a computational description as Marr presents it and so Chomsky's continued use of "computational level" to describe generative grammar is non-standard. In any case, these are not matters to be settled by appeals to authority and whether any of the proponents of the Marrian interpretation of generative grammar mentioned above share this interpretation is unclear.

To summarise these claims; unlike computational level theories, generative grammars don't describe processes, are strongly intensional, and are structure grounding. Generative theories give a procedural characterisation of a state, knowledge of language, while computational theories give a static characterisation of a process. This much shouldn't be controversial, though it is seldom acknowledged in print and it certainly isn't itself an objection to the generative method.[12] It's not surprising that language, which must be acquired and varies at least in its surface manifestations, would require a different approach from other cognitive capacities. What I will examine in the rest of this paper is whether we can give an account of generative grammar that captures the methodological virtues of the Marrian method while adhering to the constraints placed on it by the properties listed above. In the next section, I will look at several ways in which theorists have attempted to reinterpret generative grammar as a computational level theory and discuss the challenges they face before considering an alternative approach that may have a better chance (but nonetheless has problems of its own).

5. *Alternative interpretations*

5.1. *Grammars as metamathematical descriptions*

One option is to treat generative grammars as highly abstract description at the computational level. Manfred Krifka responds directly to Chomsky's claim that generative grammar does not concern input-output systems writing: "I do not see why representation levels should only be applicable to computation arising in input/output systems. In particular, one could see level 3 descriptions as idealisations, for example, the Peano axioms for our integrated arithmetic abilities, or the rules postulated by a generative grammar for our linguistic abilities" (Krifka 2011: 55). Analogies between generative grammar and Peano arithmetic are quite common and so this proposal deserves consideration.[13] The core idea seems to be that, by providing metamathematical

[12] Poeppel (2012) and Poeppel and Embick (2015) raise a range of challenges faced by attempts to connect linguistic theory and the neurobiology of language. However, they do appear to accept that traditional generative linguistics has been targeted as computational-level theorising (Poeppel 2012: 50; Poeppel and Embick 2015: 359).

[13] Chomsky (1999: 41–42), Adger and Svenonius (2015: 1422), and Boeckx (2010) all speak of the computational level as the study of the "logical properties" of the language faculty.

characterisations of the function implemented we can learn about the language faculty. Chomsky himself has made similar claims:

> One of the properties of Peano's Axioms PA is that PA generates the proof P of '2 + 2 = 4' but not the proof P of '2 + 2 = 7' (in suitable notation). We can speak freely of the property 'generable by PA' f holding of P but not P¢, and derivatively of lines of generable proofs (theorems) and the set of theorems without postulating any entities beyond PA and its properties. (Chomsky 2001: 41-42)

This excerpt has been a constant source of debate over the last two decades and I won't recapitulate the controversy here.[14] The axiom analogy does help clarify matters to an extent. Steps in a proof or derivation can be ordered and when viewed as mathematical objects there is no need to regard that order as temporal rather than structural. The connections between computation and deduction are relatively well understood and whether a set of axioms and associated rules generate a proof does not depend upon their actually being used in real-time to generate that proof.

The problem with the analogy is that it raises as many questions as it answers. The axiomatic view of grammar, while aligning with the parsing-as-deduction approach to grammar (e.g. Johnson 1989), commits us to some internal system of representation, i.e a formal language. If this is the case, then it merely pushes any question about our grasp of language back to another level (a "language of thought" needs a grammar too). The concern is that any formalism with the expressive power to represent the full range of syntactic structures found across natural languages (e.g. weak monadic second order logic) would itself require a grammar in which its combinatorial possibilities are specified. While the strength of this criticism depends on how robustly the notion of axiom is taken, most axiomatic systems require their syntax to be specified in some metalanguage.[15] Furthermore, metamathematical statements such as the Peano axioms don't tell us anything without either a set of inference rules, e.g. modus ponens, or a model. Lacking either a proof theory or a semantics, they are merely marks. It would presumably be these "logical capacities" in which we are interested when appealing to such an axiomatisation. Yet if this is the proposal, the theory doesn't support any further claims. For example, it is unwarranted to claim that a cash register is capable of inferring according to modus ponens on the grounds that it computes addition. To say that the cash register has in any sense the capacity to make logical inferences according to the rules of a classical proof system or that it can map variables to models of Peano arithmetic is simply false.

This problem is not solved by weakening our proof system (e.g. using Heyting arithmetic or intuitionist arithmetic) to get us any closer to

[14] Paul Postal described it as "the most irresponsible passage written by a professional linguist in the entire history of linguistics" (Postal 2004: 296).

[15] A discussion of the role that such concerns played in the early development of generative grammar can be found in Mallory (2023).

the "psychological reality" of the machine. A cash register is neither capable of inferring "p" from "~~p" nor incapable of it. The trivial reason is that computing the sum of x and y is not the same as deriving a proof that x + y = z. The tasks described at the metamathematical level by our formal system and at the computational-level by our function are different. Nevertheless according to Pylyshyn, "[t]his is exactly the goal Chomsky declared many years ago for linguistics: find the least powerful formalism for expressing the range of human languages and you will have a formalism that you can view as intensionally (as opposed to merely extensionally) significant" (Pylyshyn 1991: 14).

The moral here is that, when making inferences from the existence of one capacity to the existence of another we must be careful not to conflate metamathematical and algorithmic levels of description. For example, if the system in question performs multiplication using the Karatsuba algorithm, we can infer that it is capable of addition as well since the algorithm requires this. We can't however, make this kind of conclusion based on metamathematical ideas alone, e.g. the fact that recursive definitions of multiplication tend to utilise addition doesn't tell us much. In Skolem arithmetic (a complete and decidable subsystem of Peano arithmetic) multiplication is defined independently of addition. If we want to preserve the function-theoretic outlook by making the theory more abstract, it becomes less clear what the theory is actually telling us about the mind.

5.2. *Grammars as theories of performance*

5.2.1. *Grammars as parsers*

The next option is to construe grammars as computational-level descriptions of parsers. Parsing is the process of incrementally building representations of the syntactic structure of input sentences, in other words, mapping strings (one kind of information) to hierarchical structures (another kind of information). It can therefore be understood as an input-output system. Several researchers have explicitly attempted to bridge the gap between generative practice and Marrian metatheory this way. One of the most sophisticated developments of the grammar-as-parser view is provided by Neeleman and Van de Koot (2010).[16] Neeleman and Van de Koot present the minimalist operation merge as an abstract characterisation of the actual operation which a parser uses to build structural representations of sentences.

Just as Marr decomposed the algebraic properties of addition in the cash register example, Neeleman and Van de Koot discuss the abstract properties that structure-building operations might possess. Constraints on the structure-building process such as whether it is binary-

[16] A similar "one system" view is defended by Lewis and Phillips (2015). Ruth Kempson's program of Dynamic Syntax may also be seen as a higher-level description of the parser but one which takes account of the linear order of sentences (Kempson et al. 2001).

branching, inclusive, whether labels are assigned, and so forth, can be either built into the parser's structure-building process or left as filters on outputs. When these constraints are understood as properties of the parser's structure-building operation, then the computational burden of parsing is lightened significantly—fewer candidate structures are constructed and filtered. Furthermore, the properties can be characterised and discussed independently of any algorithm which implements them.

The decision to interpret merge as a parsing operation is to treat it as an aspect of performance rather than competence. Parsing a sentence is a real-time, memory-bounded cognitive process, whereas generative linguistics was initially developed as a theory of competence, a description of the state of mind corresponding to knowledge of a language, not its actual use (Chomsky 1965).[17] Whether or not you consider this reinterpretation to be a bad thing will depend upon your prior metatheoretical commitments. Nonetheless, it is worth noting some issues with this approach.

First, the operation "merge" for example, applies first to the most embedded constituent in a sentence (e.g. the "$\_\_\_\__x$" in "what$_x$ did the kitten swallow $\_\_\_\__x$?") and builds a syntactic structure outwards from this, moving constituents to higher (and more fronted) positions in the process. In English, more embedded constituents tend to appear closer the end of a clause. Parsing, in contrast, begins with the most leftward constituent in a structure and builds structure from there. As a higher-level theory of parsing, then, generative grammars start their derivation at the wrong end of the sentence. Second, the information a grammar has available to it is much richer than the information a parser has access to. Formal models of grammar assume that the inputs to the merge operation contain explicit, structured sets features which are checked off in the process of structure-building. The inputs to a grammar wear their syntactic properties on their sleeve. In contrast, the inputs to the process of human parsing are underspecified chunked phonological units or bare strings (as garden-path sentences show). This capacity for ambiguity makes parsing a difficult challenge. So if we are to view a generative theory of structure-building as a model of how the parser builds structures, it's reasonable to ask if it's likely to be a good one. Parsing is a cognitive process that lends itself well to computational-level theorising but whether the operations described by competence grammars can be translated neatly into a such a theory is open for debate (much of which is outlined in Pereplyotchik (2017)).

5.2.2. *Implementation level concerns*

[17] This is not to say that researchers weren't almost immediately attempting to connect generative claims to models of performance in work culminating in the derivational theory of complexity (Fodor and Garrett 1966; Garrett, Beaver and Fodor 1966).

Some of the most exciting contemporary research is coming from psycholinguistics. Before continuing, we should consider what the foregoing discussion means for attempts to identify the neurological correlates of the operations described by generative grammarians. In an influential body of research, Angela Friederici has assembled considerable evidence that merge occurs in the ventral part of BA44 (the posterior inferior frontal gyrus) (Friederici 2017; Liu et al. 2023). Similarly, Eliot Murphy has developed a sophisticated account of the implementation of merge, according to which the operation is realised by cross-frequency interactions between θ and ɣ frequencies where lexical items indexed by ɣ cycles are embedded within slower θ and **δ** oscillations (Murphy 2020). Others have sought the neural correlates of other operations posited by generative grammarians such as *search* (Ohta, Fukui and Sakai 2013), *label* (Murphy 2015), and *scrambling* (Makuuchi et al. 2013).

However, if we accept the claim that generative grammar does not describe performance, then there is no way to reconcile the psycholinguistic claims that the *inferior-frontal cortex,* (IFG) (Caplan et al. 1998) putatively affords core-syntactic operations such as "merge" (Zaccarella et al. 2017) and "movement" (Grodzinsky and Santi 2008; Makuuchi et al. 2013) with claims by Chomsky like the following:

> [A] generative system involves no temporal dimension. In this respect, generation of expressions is similar to other recursive processes such as construction of formal proofs. Intuitively, the proof 'begins' with axioms and each line is added to earlier lines by rules of inference or additional axioms. But this implies no temporal ordering. It is simply a description of the structural properties of the geometrical object proof. (Chomsky 2007: 6)

None of these positions are compatible with the view of merge or any other syntactic operation as an atemporal "logical" operation—logical abstractions don't show up in fMRI scans (which makes it striking that Chomsky (2017) agrees with Friederici's findings). If merge is a "logical operation" not taking place in space or time, it doesn't take place in the inferior frontal gyrus.[18] It seems reasonable to conclude that, while much of this research is exciting and important, there is little reason to believe it is tracking what generative grammarians are describing when they develop theories of syntactic structure-building *unless we reinterpret those theories as theories of parsing or some other linguistic performance.* This position has been advocated by some psycholinguists for independent reasons (Phillips 2013; Embick and Poeppel 2005).

[18] This is known as the problem of ontological commensurability (Poeppel 2017). "The tendency in generative syntax, for example, is to speak as if the computations proposed in syntactic analyses need not be regarded as computations that are performed in real-time [...] This assumption simply makes the link between linguistics and neuroscience harder to bridge, for reasons that are ultimately historical, and not necessarily principled" (Poeppel and Embick 2005: 114).

5.3. *Grammars as data-structures*

The final account we'll look at claims that generative grammars specify the data structures that are accessed by the parsing mechanism. This would place grammars at what Christopher Peacocke calls "level 1.5," somewhere between computational and algorithmic level theories. According to Peacocke, a theory at this level "identifies the information drawn upon by an algorithm" (Peacocke 1989).[19] This approach aligns with Bresnan and Kaplan's *Strong Competence Hypothesis* (the idea that a competence grammar is used by performance systems).[20] If correct, parsers are algorithms that utilise grammars to produce appropriate syntactic structures for an input string. This gives substance to the idea that a grammar "underlies and accounts for," "determines" or "is put to use by" performance (each of these phrases occur without further elaboration in Chomsky (1964)). The question then is, what is the role of a structure-building "computational" operation like merge in the theory of parsing? To be clear, we are not now considering the operations that might combine and label input constituents during parsing, but instead, we are looking at the role of an operation like merge in the grammar accessed by the parser. Parsing algorithms have their own range of real-time computational operations, e.g. pushing a unit of information to a stack, adding information to a table, searching for a representation of a rule in the grammar (see Jurafsky and Martin 2008 for introductions to basic parsing algorithms). These operations are described at the algorithmic level although one can perhaps have a computational level theory of the parser as Neeleman and van de Koot demonstrate. Typically, as algorithmic level theories, they involve representational commitments, e.g. a grammar is in Chomsky Normal Form, and so the question is how the computational operations described by a generative grammar are represented within the grammar.[21] What we are concerned with is the role of structure-building operations posited by grammarians in these accounts.

[19] Momma and Phillips also suggest that Marr's hierarchy may be best viewed as a continuum in order to accommodate the anomalous position of linguistics and neurolinguistics (Momma and Phillips 2018).

[20] Whether grammars are causally implicated in performance has been a matter of debate for decades. While many argue that grammars are causally implicated in performance (Fodor 1985; Peacocke 1986, 1989; Rey 2003; Hornstein 2009), John Collins argues that they aren't causally implicated (Collins 2017, 2023). Higginbotham claims that whether or not grammars play a causal role in production is to be determined by empirical enquiry (Higginbotham 1982). Generally, formulations of how a grammar relates to performance haven't added much detail to Chomsky's original brushstrokes, e.g. "[k]nowledge of language guides/provides the basis for actual use, but does not completely determine use" (Boeckx 2009: 134). Rey (2020) claims that a grammar makes claims about cognitive processes and architecture "at some level of abstraction," a position I think many would get behind (Rey 2020: 112).

[21] This is a considerable simplification. Designing a parser often involves deciding which rules should be represented in the grammar (i.e. a data structure separate

For example, the operations of phrase rewriting in a phrase structure grammar manifest as relations between categories when that grammar is being consulted by a parser. In the simplest example, a rule of grammar, NP → Det N, is not to be viewed as a rewrite rule for deriving some structure for another but as a statement in a database which can be accessed by the parsing algorithm. While it is sometimes suggested that grammatical operations have to be applied to generate structures so that the structure is, in effect, built twice during parsing, this isn't the role that grammar operations play in contemporary, highly-lexicalised parsing models where the structure generating information is built in to the lexical item rather than into rules relating grammatical categories. Consider Stabler's influential Minimalist parser (Stabler 2014; Berwick and Stabler 2019; Hunter 2019). While Minimalist grammars are typically "bottom-up"—derivations are formed by merging lexical items into more complex units and then merging those units until the derivation is complete, Stabler's minimalist parser is top-down, in the sense, that it starts with the highest category of a phrase along with a queue of predictions for what lies below it before applying rules operating over the input and the queue of predictions. What we need to examine is the role of merge in the grammar of this model. When we do, we see that the role of merge in this model is to specify the properties of the grammar, i.e. the sets of features associated with each lexical item. The features that lexical items are taken to have are just those that they *would* need *if* merge *were* the actual operation by which syntactic structures were built. Whether or not two items can be merged by the grammar is determined by those items syntactic features. The conceptual function of merge in the grammar is simply to induce upon the lexicon, the set of features they would require if syntactic structures were to be constructed by means of merge. But once we have the grammar on the table—the set of lexical items with their rich array of syntactic features, then we can, in effect, ignore merge when talking about how the grammar interacts with the parser. The grammar is simply a structure of the lexicon upon which parsing operations can then be defined. At no point in the model is it assumed that merge is actually implemented in the brain to generate syntactic structures. Merge is not really a computational operation at all. It doesn't describe a lower-level algorithmic process but it does give us the means to identify a set of features which such a (parsing) process may access.

Isn't this just an algorithmic level theory? Yes and no. A parser model is an algorithmic-level theory. It specifies an algorithm for computing an output for a given input in a finite series of steps. The algorithmic level theory tells us what those steps are, it specifies the algorithm,

from the parser which can be altered while keeping the parsing algorithm the same) and which rules are to be built into the operation of the parser itself. Pereplyotchik (2017) gives a helpful overview of these issues.

and in the process it makes claims about how the information must be represented. In the case of minimalist parsing models, information is represented as a set of features of lexical items. In Stabler's parsing model, this information can be understood as a structure within the lexicon. However, merge is not the computation that searches through the lexicon to incrementally build syntactic structures. This brings us to a final approach to understanding generative grammar as a computational level theory.

6. Anti-realist computational level theories

This interpretation of computation in generative grammar is both *modal* and *anti-realist* (or perhaps instrumentalist). It is modal in that it treats merge not as an operation that does apply to build syntactic structures but a computational operation that *could* apply. While it is *anti-realist* because it regards the merge-story of how structure could possibly be built as a useful theoretical device for describing how syntactic information is organised in some cognitive structure rather than a representation of an actual cognitive operation (in the style of Marr). The core idea is that, one way to describe what syntactic features lexical items need to have to be effectively learned and parsed is to describe a simple device for building syntactic structures and ask what information it would require to do its job. Then, once we know what this information is, we discard the structure-building operation. It is not posited as "cognitively real," in the sense that it doesn't pick out any real-time process. What is genuinely represented in the brain are syntactic features which are accessed by the parser. An operation like merge is a notational device for figuring out what those features might be.

If this is actually the idea that has been implicit within the literature, it would make sense of some of the stranger claims one finds in Chomsky. For example, consider the following:

> We can discuss the set of expressions or derivations generated by a grammar but in doing so no new entities are postulated in these usages beyond FL [the faculty of language], its states L [some language], and their properties. Similarly, a study of the solar system could introduce the notion HT = {possible trajectories of Halley's Comet within the solar system}, and the studies of motor organization or visual system could introduce the notions plans for moving the arm or visual images for cats (vs. bees). But these studies do not postulate weird entities apart from planets, comets, neurons, cats, and the like. (Chomsky 2001: 41-41)

Read descriptively, there are obvious problems with this analogy. Firstly, a comet does actually follow one of these trajectories—the set of trajectories is a description of paths the comet *might* take. They constitute a modal claim about the possible behaviour of the comet. In contrast, the set of derivations a grammar generates is not a description of a grammar's *possible* performance. Secondly, we do not characterise a comet as a device recursively enumerating its possible trajectories.

The trajectories of a comet are determined by things like mass and velocity, properties of the comet which constrain its possible behaviour and apply to other objects as well. Finally, it is often claimed that the outputs of grammars are involved in mediating the interface between systems of phonology and semantics. One might reasonably think that such entities would have to exist in order to do this. However, if these outputs are not generated, it is not clear how they could serve this role in transduction.

However, these objections arise only if we understand the modal component of the interpretation as a *de re* claim about an actual computational operation. If we read this section as a discussion of what the merge operation *could* do, then it seems like Chomsky is making a claim about an actual, cognitively implemented computational operation (that can be functionally localised etc.). However, if we understand it as a claim about the kind of theory that can help us uncover the structure of the lexicon, which features lexical items possess, what unpronounced items such as functional heads there might be, then it becomes a more plausible, instrumental claim about a useful kind of theory building. It is not just that the syntactic structures enumerated by a grammar are an idealisation of some cognitive structure, but the operation involved in their generation, merge, is an idealisation of this structure as well.[22] This kind of anti-realism about computation in generative linguistics has recently been advocated by John Collins: "what makes a system a computer is that only a computational theory is adequate for its explanation, independently of whether or not any physical states are discriminable as realising the computation" (Collins 2023). If this interpretation is correct, then merge is not an operation in the brain. It is merely a way of describing a data structure. It isn't just that merge doesn't occur in real time, it isn't supposed to characterise a process that does. Its function within the generative theory is to help linguists to identify syntactic features and phenomena that emerge from how syntactic information (which is actually drawn upon by the parser) is organised in the brain. Accordingly, when humans evolved the capacity for merge, they developed the capacity to organise information in their brains in such a way that, *were* merge to occur using information organised this way, it would generate the structures we find in a languages.[23]

[22] It is easy to be misled by reference to "idealization" here. In this case, merge would not be an idealisation of some actual structure building operation (considered independently of memory limitations, for example), but a theoretical tool which gives us the formal resources required to describe the functional properties of the lexical items which any such structure-building operation would have to have access to. Insomuch as it idealises actual performance processes, it does so obliquely, by enabling researchers to better describe the information that such processes have access to (i.e. as a competence theory).

[23] This is similar in a respect to Adger (2022). Adger argues that the representations posited in generative grammar are structured abstractions of brain states.

This still leaves us with a range of questions for the antirealist: why think that the relevant syntactic features are those that facilitate the operation of a structure-building device that isn't actually responsible for building syntactic structures in real time? Why aren't the features we posit the ones that are directly posited to facilitate efficient parsing (as in Ruth Kempson's Dynamic Syntax program)? If reference to merge is an expression that one is adopting a distinct framework of idealisation, how should we think of the empirical content of generative theories as well as theoretical debates about the exact nature of merge (e.g. binary merge, parallel merge, workspace models)? These are important questions for anyone who adopts the generative framework as it has been described here and it is far from certain that they have easy answers. For some, I suspect that this position will be too great a concession to abstraction.

The present paper has merely sought to illustrate that generative grammar is not a computational level theory in the traditional Marrian sense assumed throughout much of computational cognitive neuroscience and suggest that researchers in psycholinguistics are unlikely to find the structure-building operations discussed by linguists in their labs. I have also suggested that a more instrumentalist interpretation may make sense of how generative "computations" are appealed to within parsing theory. Throughout this, I have tried to balance both empirical, theoretical, and to some extent, hermeneutic evidence. The practitioners of a scientific discipline are by no means obligated to interpret their own research in accordance with the ideas and images of prominent figures within their field. The fact that Chomsky might interpret the subject matter of generative linguistics in a certain way should not bind others in the field. I do, however, think there is value in being explicit about the promises and challenges of different metatheoretical commitments, in particular due to the interdisciplinary nature of current research.

Acknowledgement

The bulk of this paper was written long ago and benefitted from discussion with David Ager and Mark Textor. More recent additions have benefited from comments at the OMLET in Oslo, in particular from Drew Johnson and Nicholas Allot, and in Durham from James Miller, Aadil Kurji and Keith Begley.

References

Abels, K. 2013. “Comments on Hornstein.” *Mind & Language* 28 (4): 421–429.

Adger, D. and Svenonius, P. 2015. “Linguistic Explanation and Domain Specialization: A Case Study in Bound Variable Anaphora.” *Frontiers in Psychology* 6: 421.

Adger, D. 2021. “The Architecture of Computation” In N. Allot, T. Lohndal and G. Rey (eds.). *A Companion to Chomsky*. Hoboken: John Wiley And Sons, 123–139

Adger, D. 2022. “What are Linguistic Representations?” *Mind & Language* 37 (2): 248–260.

Anderson, J. R. 1990. *The Adaptive Character of Thought*. Hillsdale: Lawrence Erlbaum Associates.

Berwick, R. C. 1985. *The Acquisition of Syntactic Knowledge*. Cambridge: MIT Press.

Berwick, R. C. and Chomsky, N. 2016. *Why Only Us?* Cambridge: MIT Press.

Berwick, R. C., Pietroski, P., Yankama, B. and Chomsky, N. 2011. “Poverty of the Stimulus Revisited.” *Cognitive Science* 35 (7): 1207–1242.

Berwick, R. C. and Stabler, E. P. (eds.). 2019. *Minimalist Parsing*. Oxford: Oxford University Press.

Boeckx, C. 2010. *Language in Cognition*. Oxford: Wiley-Blackwell.

Chomsky, N. 1965. *Aspects of the Theory of Syntax*. Cambridge: MIT Press.

———. 1980. “Rules and Representations.” *The Behavioural and Brain Sciences* 3 (127): 1–61.

———. 1982. *Some Concepts and Consequences of the Theory of Government and Binding*. Cambridge: MIT Press.

———. 1995. “Language and Nature.” *Mind* 104 (413): 1–61.

———. 2001. “Derivation by Phase.” In M. Kenstowicz (ed.). *Ken Hale: A Life in Language*. Cambridge: MIT Press, 1–52.

———. 2007. “Approaching UG from Below.” In U. Sauerland and H. Gärtner (eds.). *Interfaces + Recursion = Language?: Chomsky’s Minimalism and the View from Syntax-Semantics*. Berlin: Mouton de Gruyter, 1–29.

———. 2013. “Problems of Projection.” *Lingua* 130: 33–49.

Chomsky, N., Seely, T. D., Berwick, R. C., Fong, S., Huybregts, M. A. C., Kitahara, H. and Sugimoto, Y. 2023. *Merge and the Strong Minimalist Thesis*. Cambridge: Cambridge University Press.

Collins, J. 2017. “Faculties and Modules: Chomsky on Cognitive Architecture.” In J. McGilvray (ed.). *The Cambridge Companion to Chomsky*. Cambridge: Cambridge University Press, 217–234.

———. 2023. “Generative Linguistics: ‘Galilean Style’.” *Language Sciences* 100: 101585.

Egan, F. 2017. “Function-Theoretic Explanation and the Search for Neural Mechanisms.” In D. M. Kaplan (ed.). *Integrating Mind and Brain Science: Mechanistic Perspectives and Beyond*. Oxford: Oxford University Press, 145–163.

Embick, D. and Poeppel, D. 2005. “Defining the Relation Between Linguistics and Neuroscience.” In A. Cutler (ed.). *Twenty-first Century Psycholinguistics: Four Cornerstones*. Mahwah, NJ: Lawrence Erlbaum Associates, 103–118.

———. 2015. "Towards a Computational(ist) Neurobiology of Language: Correlational, Integrated and Explanatory Neurolinguistics." *Language, Cognition and Neuroscience* 30 (4): 357–366.

Etienne, A. S. and Jeffery K. J. 2004. "Path Integration in Mammals." *Hippocampus* 14 (2): 180–192.

Fodor, J. A. and Garrett, M. 1967. "Some Syntactic Determinants of Sentential Complexity." *Perception & Psychophysics* 2 (7): 289–296.

Fodor, J. 1985. "Some Notes on What Linguistics is About." In J. J. Katz (ed.). *The Philosophy of Linguistics*. Oxford: Oxford University Press, 146–160.

Friederici, A. D. and Kotz. S. A. 2003. "The Brain Basis of Syntactic Processes: Functional Imaging and Lesion Studies." *Neuroimage* 20 (1): S8–S17.

Friederici, A. D. 2017. *Language in Our Brain: The Origins of a Uniquely Human Capacity*. Cambridge: MIT Press.

Friederici, A. D., Chomsky, N., Berwick, R. C., Moro, A. and Bolhuis, J. J. 2017. "Language, Mind and Brain." *Nature Human Behaviour* 1 (10): 713–722.

Garrett, M., Bever, T. and Fodor, J. 1966. "The Active Use of Grammar in Speech Perception." *Perception & Psychophysics* 1 (1): 30–32.

Grodzinsky, Y. and Santi, A. 2008. "The Battle for Broca's Region." *Trends in Cognitive Sciences* 12 (12): 474–480.

Graf, T. 2017. "A Computational Guide to the Dichotomy of Features and Constraints." *Glossa: a Journal of General Linguistics* 2 (1): 18.

Heinz, J. 2011. "Computational Phonology Part I: Foundations." *Language and Linguistics Compass* 5 (4): 140–152.

Higginbotham, J. 1982. "Noam Chomsky's Linguistic Theory." *Social Research* 49 (1): 143–157.

———. 1991. "Remarks on the Metaphysics of Linguistics." *Linguistics and Philosophy* 14 (5): 555–66.

Hornstein, N. 2009. *A Theory of Syntax*. Cambridge: Cambridge University Press.

———. 2013. "Three Grades of Grammatical Involvement: Syntax from a Minimalist Perspective." *Mind & Language* 28 (4): 392–420.

Hornstein, N. and Pietroski, P. 2009. "Basic Operations: Minimal Syntax-Semantics." *Catalan Journal of Linguistics* 8 (1): 113–139.

Hunter, T. 2019. "Left-corner Parsing of Minimalist Grammars." In R. C. Berwick and E. P. Stabler (eds.). *Minimalist Parsing*. Oxford: Oxford University Press, 125–158.

Jackendoff, R. 1988. "Topic…Comment: Why are They Saying These Things about Us?" *Natural Language & Linguistic Theory* 6 (3): 435–442.

———. 2012. "Language as a Source of Evidence for Theories of Spatial Representation." *Perception* 41 (9): 1128–1152.

Johnson, M. 1989. "Parsing as Deduction: The Use of Knowledge of Language." *Journal of Psycholinguistic Research* 18 (1): 105–128.

Jonas, E. and Kording, K. 2017. "Could a Neuroscientist Understand a Microprocessor?" *PLoS Computational Biology* 13 (1): e1005268.

Johnson, K. 2015. "Notational Variants and Invariance in Linguistics." *Mind & Language* 30 (2): 162–186.

Johnson, M. 2016. "Marr's Levels and the Minimalist Program." *Psycho-*

nomic Bulletin & Review 24 (1): 171–174.

Kay. P. and Fillmore, C. J. 1999. "Grammatical Constructions and Linguistic Generalizations: The What's X doing Y? Construction." *Language* 75 (1): 1–33.

Katz, J. 1981. *Language and Other Abstract Objects*. Oxford: Basil Blackwell.

Katz, J. and Postal, P. 1991. "Realism vs. Conceptualism in Linguistics." *Linguistics and Philosophy* 14 (5): 515–554.

Kempson, R. Meyer-Viol, W. and Gabbay, D. 2001. *Dynamic Syntax: The Flow of Understanding*. Oxford: Blackwell.

Kobele, G. 2012. "Ellipsis: Computation of." *Wiley Interdisciplinary Reviews* (3): 411–418.

Krifka, M. 2011. "In Defence of Idealizations: A Comment on Stokhof and van Lambalgen." *Theoretical Linguistics* 37: 51–62.

Lewis, S. and Philips, C. 2015. "Aligning Grammatical Theories and Language Processing Models." *Journal of Psycholinguistic Research* 44: 27–46.

Liu, Y., Gao, C., Friederici, A. D., Zaccarella, E. and Chen, L. 2023. "Exploring the Neurobiology of Merge at a Basic Level: Insights from a Novel Artificial Grammar Paradigm." *Frontiers in Psychology* 14: 1151518.

Michiru, M. and Friederici, A. D. 2013. "Hierarchical Functional Connectivity between the Core Language System and the Working Memory System." *Cortex* 49 (9): 2416–2423.

Mallory, F. 2023. "Why is Generative Grammar Recursive?" *Erkenntnis* 88: 3097–3111.

Marr, D. 1982. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. New York: Freeman.

Michel, C. 2023. "Scaling up Predictive Processing to Language with Construction Grammar." *Philosophical Psychology* 36 (3): 553–579.

Momma, S. and Phillips, C. 2018. "The Relationship between Parsing and Generation." *Annual Review of Linguistics* 4: 233–254.

Murphy, E. 2015. "Labels, Cognomes, and Cyclic Computation: an Ethological Perspective." *Frontiers in Psychology* 6: 715.

Murphy, E. 2020. *The Oscillatory Nature of Language*. Cambridge: Cambridge University Press.

Neeleman, A. 2013. "Comments on Pullum." *Mind & Language* 28 (4): 522–531.

Neeleman, A. and van de Koot, H. 2010. "Theoretical Validity and Psychological Reality of the Grammatical Code." In M. Everaert, T. Lentz, H. De Mulder, H. Nilsen and A. Zondervan (eds.). *The Linguistics Enterprise*. Amsterdam: John Benjamins, 183–212.

Ohta, S. and Sakai, K. L. 2017. "Computational Principles of Syntax in the Regions Specialized for Language: Integrating Theoretical Linguistics and Functional Neuroimaging." In N. Fukui (ed.). *Merge in the Mind-Brain*. New York: Routledge, 237–264.

Peacocke, C. 1986. "Explanation in Computational Psychology: Language, Perception and Level 1.5." *Mind and Language* 1 (2): 101–123.

———. 1990. "When is a Grammar Psychologically Real." In A. George and N. Chomsky (eds.). *Reflections on Chomsky*. Oxford: Basil Blackwell, 111–130.

Pereplyotchik, D. 2017. *Psychosyntax: The Nature of Grammar and its*

Place in the Mind. Springer Cham.

Perfors, A. 2017. "On Simplicity and Emergence." *Psychonomic Bulletin & Review* 24 (1): 175–176.

Perrone, J. A. and Krauzlis, R. J. 2008. "Vector Subtraction Using Visual and Extraretinal Motion Signals: A New Look at Efference Copy and Corollary Discharge Theories." *Journal of Vision* 8 (14): 1–14.

Philips, C. 2013. "Parser Grammar Relations: We don't understand everything twice." In M. Sanz, I. Laka and M. K. Tanenhaus (eds.). *Language Down the Garden Path: The Cognitive and Biological Basis for Linguistic Structure*. Oxford: Oxford University Press, 1–23.

Piantadosi, S. 2023. "Modern Language Models Refute Chomsky's Approach to Language." *Lingbuzz Preprint* url: https://lingbuzz.net/lingbuzz/007180.

Poeppel, D. 2017. "The Influence of Chomsky on the Neuroscience of Language." In J. McGilvray (ed.). *The Cambridge Companion to Chomsky*. Cambridge: Cambridge University Press, 155–174.

Postal, P. 2004. *Skeptical Linguistic Essays*. Oxford: Oxford University Press.

Pullum, G. 2009. "Computational Linguistics and Generative Linguistics: The Triumph of Hope over Experience." *ILCL 09 Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?* 12–21.

Pylyshyn, Z. W. 1973. "The Role of Competence Theories in Cognitive Psychology." *Journal of Psycholinguistic Research* 2 (1): 21–50.

Pylyshyn, Z. 1991. "Rules and Representations: Chomsky and Representational Realism." In A. Kashir (ed.). *The Chomskian Turn*. Cambridge: Blackwell, 231–251.

Rey, G. 2003. "Chomsky, Intentionality, and a CRRT." In L. M. Anthony and N. Hornstein (eds.). *Chomsky and his Critics*. Oxford: Blackwell, 105–139.

———. 2020. *Representation of Language: Philosophical Issues in a Chomskyan* Linguistics. Oxford: Oxford University Press.

Roberts, I. 2014. "The Mystery of the Overlooked Discipline: Modern Syntactic Theory and Cognitive Science." *Revue Roumaine de Linguistique* 58: 151–178.

Schneider, A. and Mores, R. 2013. "Fourier-Time-Transformation (FTT), Analysis of Sound and Auditory Perception." In R. Bader (ed.). *Sound - Perception - Performance. Current Research in Systematic Musicology*. Vol 1. Heidelberg: Springer, 299–329.

Stabler, E. P. 2013. "Two Models of Minimalist, Incremental Syntactic Analysis." *Topics in Cognitive Science* 5 (3): 611–633.

van Rooij, I., Wright, C., Kwisthout, J. and Wareham, T. 2018. "Rational Analysis, Intractability, and the Prospects of 'As if'-explanations." *Synthese* 195 (2): 491–510.

Yang, C. 2017. "Rage Against the Machine: Evaluation Metrics in the 21st Century." *Language Acquisition: A Journal of Developmental Linguistics* 24 (2): 100–125.

Zaccarella, E., Schell, M. and Friederici, A. D. 2017. "Reviewing the Functional Basis of the Syntactic Merge Mechanism for Language: A Coordinate-based Activation Likelihood Estimation Meta-analysis." *Neuroscience & Biobehavioral Reviews* 80: 646–656.