# Theoretical Validity and Psychological Reality of the Grammatical Code[*]

Ad Neeleman and Hans van de Koot – UCL

## 1 Ristad's problem

Much progress in theoretical linguistics has been made possible by the distinction between competence and performance, which is essentially an idealization over the data that linguists work with. Chomsky (1965:3) formulates it as follows:

(1) Linguistic theory is concerned primarily with an ideal speaker-listener, in a completely homogeneous speech-community, who knows its language perfectly and is unaffected by such grammatically irrelevant conditions as memory limitations, distractions, shifts of attention and interest, and errors (random or characteristic) in applying his knowledge of the language in actual performance.

The idealization proposed here is hardly controversial: in all empirical sciences it is customary to ignore noise in the data.

A competence theory must of course be supplemented by a characterization of its relation to the systems of performance. Chomsky (1965:9) argues that the relation between the grammar and the systems of comprehension and production can be rather abstract (see also Newmeyer 2003):

(2) When we say that a sentence has a certain derivation with respect to a particular generative grammar, we say nothing about how the speaker or hearer might proceed, in some practical or efficient way, to construct such a derivation. These questions belong to the theory of language use – the theory of performance.

This view certainly goes beyond the common practice of ignoring noise in the data in insisting that the theory of competence does not necessarily tell us anything about what happens in comprehension and production. While Chomsky's quote in (2) is nearly forty years old, it is an accurate description of the current state of affairs: there is no transparent relation between minimalist transformational grammar and plausible models of human language computation. For example, language computation is presumably left-to-right, while grammar – if taken to be derivational at all – is cyclic. In conjunction with widely held assumptions about phrase structure, this implies that derivations proceed largely right-to-left.

Although we accept both aspects of the competence-performance distinction, we think that Ristad (1993) is justified in pointing out that this distinction threatens the coherence of the generative enterprise. If competence theory has nothing to say about the computations performed by speakers and hearers, one may well ask what object the theory describes (see especially Ristad 1993: 110-117 for discussion). In what follows we will refer to this question as Ristad's problem:

(3) *Ristad's problem*
What object is a competence grammar a theory of, given that it abstracts away from language computation?

[*] We would like to thank Peter Ackema, Michael Brody, Dirk Bury, Janet Dean Fodor, Vieri Samek-Lodovici, Øystein Nilssen, Neil Smith, Michal Starke, Peter Svenonius, Kriszta Szendrői, and the participants of TiLT 2004 and of the 2003 Current Issues in Syntax class at UCL for useful comments.

The main aim of this paper is to compare two answers to this problem. According to the first, the grammar is a knowledge base consulted by the performance systems; according to the second, the grammar is a description of the language faculty at a more abstract level than that at which the performance systems are best described. We will argue in favour of the latter view.

## 2 Two answers

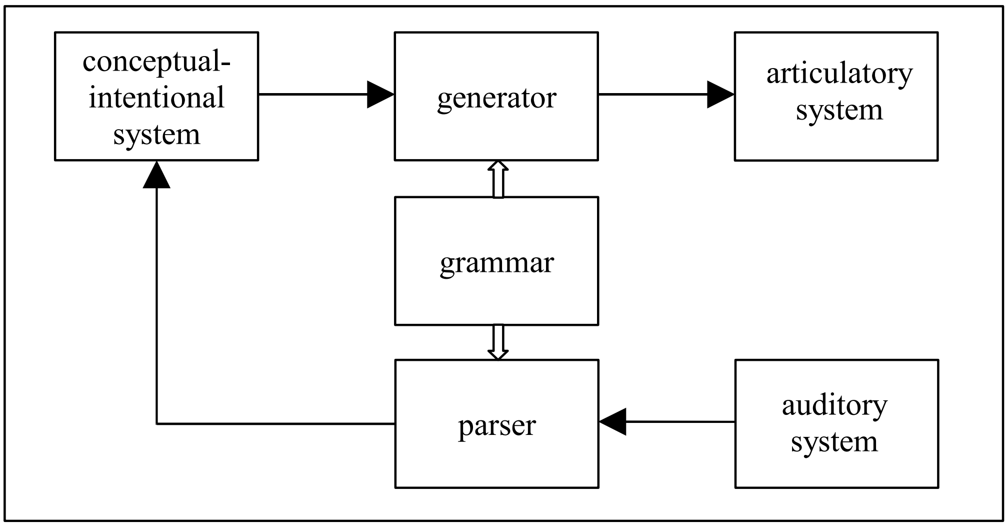
There are a number of tasks that need to be carried out before a speaker can utter, or write down, a sentence (see Levelt 1989 for discussion and further references; see Dennett 1992 for some critical remarks). We will refer to the system responsible for these tasks as the *generator*; it maps conceptual-intentional information to a set of instructions for the system of articulation (abstracting away from written or signed languages). Similarly, there are various computations that must take place before someone who hears a sentence can understand what is being said (see Pritchett 1992, Gorrell 1995, and Frazier and Clifton 1996 for discussion and further references). The system responsible for these computations, usually referred to as the *parser*, maps an auditory signal to conceptual-intentional information.

One might argue that the grammar is a separate module of the brain consulted in some fashion by the performance systems. Chomsky (2000:117) seems to suggest an architecture of this type:

(4) There is good evidence that the language faculty has at least two different components: a "cognitive system" that stores information in some manner, and performance systems that make use of this information for articulation [and] perception […].

This quote does not give an explicit characterization of the relation between competence and performance. One model compatible with it is given in (5), where the double arrows are meant to represent the information flow from grammar to parser and generator.

(5)



This model provides a straightforward justification for Chomsky's idealization in the quote in (2) and thereby suggests an answer to Ristad's problem. Generative linguistics is the study of the knowledge module at the center of (5). Linguistic behavior is the result of the interaction of this module with the generator and parser. Hence, if one wants to find out about the linguistic module, one should abstract away from various aspects of this behavior.

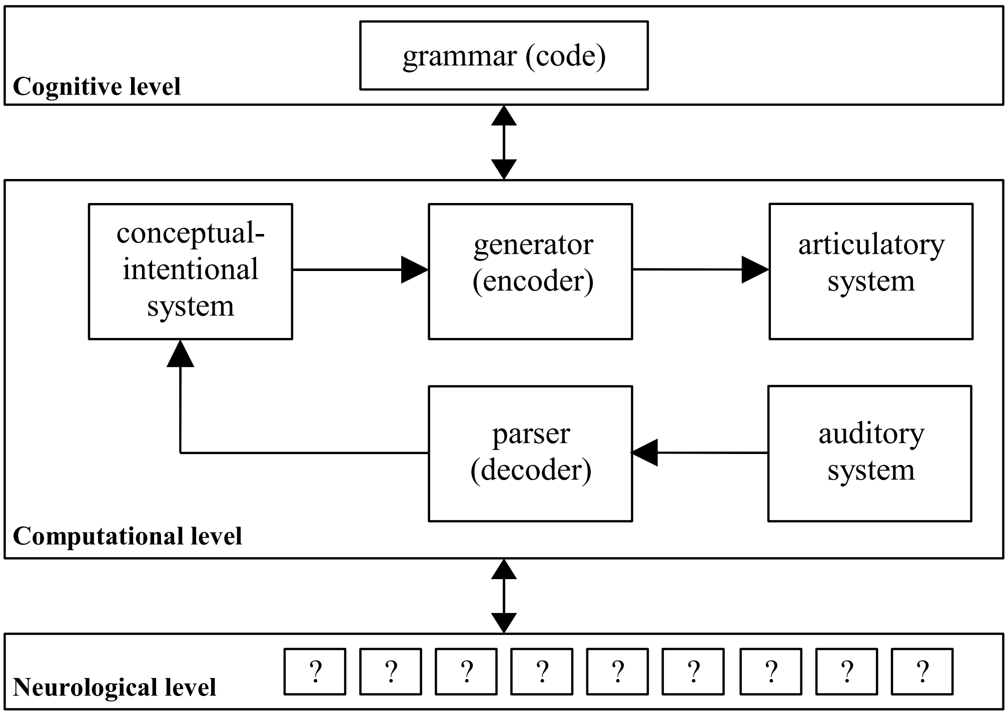
An alternative to the view that the grammar is a knowledge module at the same level of description as the performance systems is suggested by Marr's (1982) discussion of complex information-processing systems. Marr argues that any machine carrying out an information-processing task must be understood at three levels. The most abstract level is a description of the device in terms of the logical structure of the mapping that it carries out from one type of information to another. In the case of humans, we can refer to this as the cognitive level of description. And in the case of language it is the natural locus of hypotheses about regularities in the mapping between sound and meaning.

The next level down is a description of the algorithm that yields the desired input-output mapping. We will call this the computational level.[1] Of course, the representations associated by the algorithm must stand in a relation of logical equivalence (or something approaching it) to the descriptions at the cognitive level (see Abney 1988 and Van de Koot 1990 for discussion). But the algorithm itself may bear little resemblance to the mathematical characterization of the mapping at the cognitive level. In the case of language the computational level consists of a description of the performance systems.

At the most basic level, we must offer a description of how the algorithm and its input and output are realized physically. In the case of human language, this is a theory of the neurology associated with linguistic computation.

The following model depicts this tripartite description of the human language faculty. The generator and parser are modules at the computational level, responsible for encoding what a speaker wants to say and for decoding what a hearer receives as input. The grammar is a description at the cognitive level of the logical structure of these computations, a kind of linguistic code.

(6)



The main difference between the models in (5) and (6) is that the former assumes that the theories of competence and performance characterize two objects at the same level of description. This must be the case, because the model crucially assumes information flow between the grammar and the parser/generator. In constrast, the model in (6) assumes that the theories of competence and performance characterize a single object at different levels of description. Both proposal constitute answers to Ristad's problem, but with very different implications.

[1] What we call the cognitive level corresponds to Marr's computational theory. What we call the computational level corresponds to Marr's algorithmic level.

Those readers only familiar with recent work in transformational grammar – the so-called minimalist program – may feel that the model in (6) is a dramatic departure from generative orthodoxy. In fact the opposite is true. Marr himself points out that Chomsky's theory of transformational grammar belongs to the most abstract level of description. As he puts it (Marr 1982: 28):

(7) [F]inding algorithms by which Chomsky's theory may be implemented is a completely different endeavor from formulating the theory itself. In our terms, it is a study at a different level, and both tasks have to be done.

One might think that developments since the publication of *Aspects of the Theory of Syntax* have invalidated this view, an impression strengthened by the way in which syntacticians often talk about derivations as actual computations and of the syntax as the computational system. Consider, for instance, the motivation for the notion of 'phase' (comparable to the more traditional notion of cycle), which plays an important role in current minimalist thinking. As Chomsky (2004) makes clear, phases are assumed to limit the search space for linguistic operations such as movement, thereby reducing the load on short-term memory.

Outside the derivational tradition in linguistics, the model in (6) is probably much less controversial. Many psychologists, for example, would agree that the grammar is not a knowledge base consulted in actual language computation. However, those who hold this view often tie it to an instrumentalist interpretation of grammar, according to which grammatical theory is not about anything real, but at best a handy way of talking about the performance systems. In fact, there are also several linguists who have relativized the importance of grammatical theory in this way (see Phillips 1996, Li 1997, Kempson et al. 2001, Culicover and Nowak 2003, among others).

In what follows we will combine a defence of the model in (6) with an argument for a realist view of grammar. Generalizations that hold at the cognitive level of description are of critical importance to understanding linguistic computation (and vice versa). Before elaborating on these issues, we will argue that the model in (5) simply cannot work.

## 3 On grammar consultation

An evaluation of the model in (5) requires a viable characterization of the consultation relation between grammar and performance systems. Given that the parser and the generator build linguistic representations, we must ask ourselves what the role of the grammar is in this process. For the idea of consultation to have empirical content, it must be the case that the performance systems are capable of generating structures that the grammar rejects. If the performance systems only build structures that adhere to the grammar, then there is no plausible function left for a separate grammatical module at the same level of description.

The simplest conceptualization compatible with these requirements treats parser and generator as minimally specified structure-builders, possibly enriched with a number of task-specific strategies. The outputs of these structure builders form the input to the grammar, whose principles act as filters that either reject the input as ill-formed or pass it on to language-external systems. This proposal can be summarized as in (8a) below.

There are alternatives in which the notion of grammar consultation has progressively less empirical content. These can be derived from (8a) by interpreting some principles of grammar as part of the structure-building mechanisms, rather than as filters (see (8b)). If all principles are reinterpreted in this way (see (8c)), it is fair to say that grammar consultation is entirely redundant, as parser and generator are unable to build structures that violate the grammar.

(8) a. All principles of grammar are filters on structures proposed by parser and generator.
    b. Some principles are built into parser and generator, other principles are filters.
    c. All principles are built into parser and generator.

A reasonable interpretation of (8a) assumes a parser that is limited to binary branching structures and that obeys feature cooccurrence restrictions (such as the restriction that no node can belong to two different categories). Such a parser can generate a very large number of structures for even simple inputs. For a three-word input, the number of trees will be equal to $2c^2$, where $c$ is the number of possible feature combinations that can characterize a node. This complexity has two sources, namely tree geometry and the content of nodes. The parser has a choice of two tree shapes (this yields the factor 2). In addition, a minimal tree for a three-word sentence contains two complex nodes, each of which can have $c$ feature combinations (this yields the factor $c^2$). The number of possible trees grows very rapidly with every additional word, since the rate of growth in the number of tree geometries is almost factorial, while the rate of growth in the number of node content assignments is exponential (given 7 mutually compatible features, $c$ equals 128, which implies that there are $5*128^4$ potential trees – around $13*10^8$ – for an input string of 4 words).

While the set-up just sketched is a strawman, it allows us to see that a simple generate-and-test design yields so many candidate structures that any computational system is quickly overwhelmed by it. There are three main strategies for addressing this issue. (These are not mutually exclusive and in practice researchers typically employ more than one of them.) The first is to further constrain the parser, so that fewer candidate structures can be built. The second is to call on filters as early as possible rather than at the end of the structure-building process. Finally, one may be able to limit the number of candidate structures by underspecifying them. We discuss the merits of each of these strategies in turn, limiting ourselves to the parser (all conclusions we draw carry over straightforwardly to the generator).

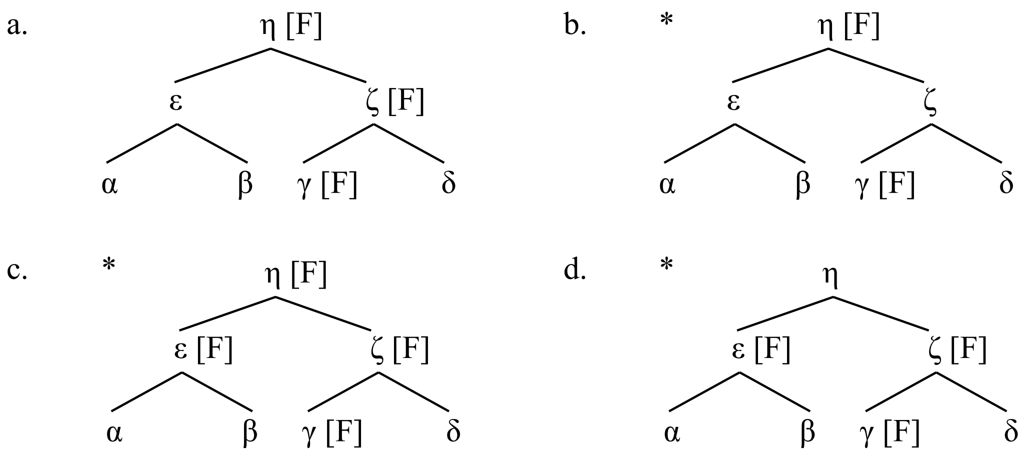
## 4. The benefits of grammar compilation

The transfer of principles from grammar to parser can only be beneficial if the computational problems of the original generate-and-test design are not replicated internally to the parser. That is to say, this module should not itself consist of an unconstrained structure assigner plus a set of filters (for discussion of this point in the context of Government and Binding Theory, see Kolb and Thiersch 1991). Therefore, many researchers adopt an approach in which inputs are pre-parsed using a set of context-free rewrite rules derived from various principles of the competence grammar. (Indeed, parsing with such a system requires no more than cubic time; see Younger 1967 and Earley 1970.) The structures recovered in this way are then presented to a set of filters that correspond to other principles of the competence grammar. This strategy of translating grammatical principles into low-level rules is usually referred to as grammar compilation (see Marcus 1980, Berwick & Weinberg 1984 , and Abney 1989, 1991).

Grammar compilation has advantageous computational effects because it eliminates the search space associated with very general and abstract constraints by recoding them in a (possibly large) number of locally applicable rules. Let us illustrate this using Inclusiveness, the main principle regulating the content of nodes in current theories of syntax (Chomsky 1995a,b; the formulation below is taken from Neeleman and Van de Koot 2002):

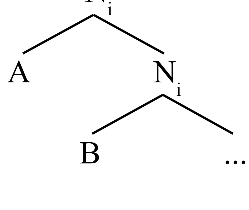
(9) *Inclusiveness*
Properties of a nonterminal node must be recoverable from its daughters; properties of a terminal node must be recoverable from the lexicon.

Inclusiveness allows upward transfer of features, as in (10a), as long as no node is skipped, as does happen in (10b). It does not allow downwards or sidewards transfer of features, as in (10c) and (10d).

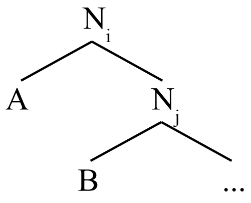
(10)


The effects of Inclusiveness can be encoded in a context-free grammar if every feature that is part of the left-hand side of a rewrite rule recurs in the right-hand side of the rule. We illustrate how this can help reduce computational load in parsing using a toy example. Consider a hypothetical language whose grammar can be characterized as follows: (i) trees are uniformly right-branching binary structures; (ii) the feature content of nodes is governed by Inclusiveness; (iii) percolation of features can only take place from the right; (iv) all features must percolate to the root node; (v) a node can contain one of *c* possible feature combinations.

Consider first a set-up in which the above conditions on the percolation of features are built into the parser in the form of a set of context-free rewrite rules that emulate feature inheritance from right daughters (the general format would be $N_i \rightarrow X\ N_i$, where $N_i$ is any node with a well-formed feature specification). Given that the number of possible feature combinations is *c*, the compiled grammar will consist of a maximum of $c^2$ rewrite rules (namely the number of possible values for X multiplied by the number of possible values for $N_i$). Suppose we are parsing the string *ABC* and have not yet encountered C (see the partial structure in (11)). The maximum number of trees that the parser will need to consider upon parsing C is *c*. Once *A* has been found, the parser can postulate *c* alternative partial structures, given that its mother can be the left-hand side of only *c* rewrite rules. Once *B* has been found, the parser can expand each of these partial structures in exactly one way, because *B*'s mother and *A*'s mother are guaranteed to be identical in feature content (in (11) this is indicated by the index *i*). Upon encountering C, the parser only needs to determine which of the *c* possible feature combinations is contained in that terminal. Therefore, irrespective of the length of the input string, the search space for the parser never exceeds *c*. In conclusion, this very restricted grammar is not associated with any serious computational load, if compiled in the manner outlined above.

(11)


Suppose instead that Inclusiveness and right-headedness are filters and that the parser builds right-branching structures only. This arrangement leaves the parser free to postulate any permissible feature combination for any node; illegal patterns of percolation are only filtered out once a complete tree is presented to the grammar. If this parser analyzes the string *ABC*, the number of trees to be considered upon encountering the final input symbol will be $c^2$. This is because nothing in the parser links the feature content of the nodes that immediately dominate *A* and *B* (see (12)). In general, the parser's search space at the end of a parse will be $c^{n-1}$, where *n* is the number of symbols in the input string. This growth rate is the result of a generate-and-test design in which Inclusiveness does not directly restrict the options of the parser but is only called upon when a complete structure has been built.

(12)


The reductions in search space associated with grammar compilation are illustrated in (13), which gives an overview of the number of structures assigned to strings using increasingly constrained parsers.[2] The table is based on a grammar that is more realistic than the toy example just considered. It allows projection from either the left or the right and a mixture of left- and right-branching structures. Hence, trees can be distinguished on the basis of tree shape, as well as the content of nodes. In order to keep the example workable, we assume that each lexical item has a categorial feature, and an index that uniquely identifies it. Feature percolation is restricted to projection of this information.

Column A shows how many structures the parser allows for an input string if constrained by binary branching and a weak form of Inclusiveness that allows feature percolation to skip nodes, as in (10b), but disallows sideward or downward transfer. The results in column B obtain if the assignment of a label to a nonterminal node takes place under direct domination, as required by the strict version of Inclusiveness in (9). Although this represents a marked improvement, the order of growth in column B is still unacceptable, as it outstrips that of $2^n$ in column D, included here as a benchmark. Adding directionality of label assignment to the system leads to a further decrease in the number of alternative structures (although the growth in column C is still exponential). The dramatic improvement shown in column E is a consequence of introducing selection as a further restriction on structure assignment.

(13)

| | *A* | *B* | *C* | *D* | *E* |
|---|---|---|---|---|---|
| Men (1) | 1 | 1 | 1 | 2 | 1 |
| Men slept (2) | 2 | 2 | 1 | 4 | 1 |
| The man slept (3) | 12 | 8 | 2 | 8 | 1 |
| Men bought a book (4) | 112 | 40 | 5 | 16 | 1 |
| A man bought a book (5) | 1360 | 224 | 14 | 32 | 1 |
| Men said men bought a book (6) | 19872 | 1344 | 42 | 64 | 1 |
| Men said a man bought a book (7) | ?[3] | 8448 | 132 | 128 | 1 |

A: merge with weak Inclusiveness and binary branching
B: merge with Inclusiveness and binary branching
C: merge with Inclusiveness, binary branching and label assignment under directionality
D: $2^n$
E: merge with Inclusiveness, binary branching, label assignment under directionality and selection

[2] These numbers have been calculated using a Prolog program containing a parameterized merge predicate. This program can be downloaded at www.phon.ucl.ac.uk/home/hans/pubs.html.
[3] The question mark in this cell is due to limitations of the computers available to us.

The constraints added to the parser's structure-building device in (13) correspond to well-known grammatical restrictions. Label assignment under direct domination yields the merge operation standardly adopted in minimalism (see Chomsky 1995a,b). Directionality of label assignment mimics the effects of directionality restrictions, such as those encoded by word order parameters. Selection, finally, reflects the existence of argument structure.

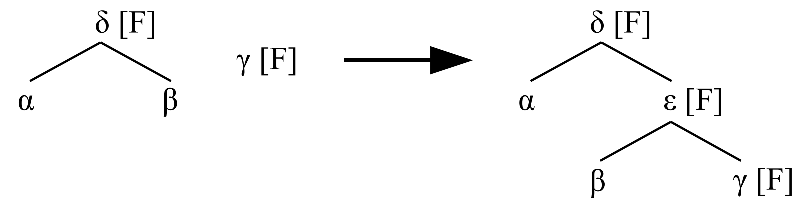
It is, in fact, widely acknowledged that efficiency in parsing can be improved by grammar compilation (see Berwick & Weinberg 1984, Fong 1991 and the papers in Berwick et al. 1991). If compilation is necessary, the question arises what the status is of compiled principles. Once a principle has been compiled, it need not be present in the grammar as a filter, because no candidate structures can be generated that violate it. This conclusion of course bears on the validity of the model in (5): the more principles are compiled, the less support there is for the view that the grammar is a module at the same level of description as the parser. Hence, proponents of (5) must find other ways of making parsing efficient.

## 5. The benefits of interleaving

The inefficiency of grammar consultation could be alleviated by allowing intermediate structures to be presented to the grammar, so that unwanted structures are eliminated early on. This technique is often referred to as 'interleaving'. The crucial question in the current context is whether interleaving can be an alternative to compilation. There are circumstances in which this is the case, such as the toy example considered earlier (we will not demonstrate this here). In general, however, interleaving cannot remove all sources of computational intractability.

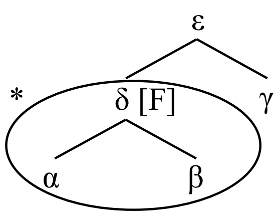
The key obstacle is that almost all intermediate structures are ungrammatical. For example, a partial structure for a transitive sentence that lacks an object violates the θ-criterion. Hence, this principle cannot be checked until all arguments have been integrated into the tree.

A problem of this type also arises in the evaluation of partial structures for adherence to Inclusiveness. Inclusiveness cannot be checked if the direct domination relations in the right edge of a tree might be altered when the parser extends the structure. A feature F that threatens to violate Inclusiveness in a node N may turn out to have to its origin in an as yet unintegrated terminal or partial tree that may end up as a daughter of N. Hence, there might be a grammatical continuation of the parse, even though the initial tree seems to violate Inclusiveness. (14) illustrates this scenario, where both β and γ might be complex categories.

(14)


It follows that Inclusiveness can be applied to a partial structure only if it is the left daughter of a larger structure whose right daughter contains at least one terminal node. This guarantees that the right edge of the partial structure in question, consisting of δ and β in (15), cannot undergo further modification.

(15)



We may conclude that, as long as direct domination relations cannot be fixed at intermediate stages of the parse, there will be cases in which Inclusiveness cannot be interleaved with structure-building operations. This gives rise to unacceptable complexity associated with node content, even if interleaving of other principles succeeds in eliminating the complexity associated with tree shape. In particular, no right-branching structure can be evaluated with respect to Inclusiveness unless one of two conditions is met. Either the parser has reached the end of the input string or it hypothesizes that the right-branching structure becomes part of a left-branching structure. But this means that in the worst case the number of candidate trees that are presented to the grammar is $O(c^n)$, where, as before, $c$ is the number of possible feature combinations and $n$ is the number of terminals dominated by the right-branching (sub-)tree. If $c$ equals 128, this implies that upon integration of the final terminal of the string *I like old port*, well over 6 million candidate structures must be considered by Inclusiveness. This example suggests that the beneficial effects of interleaving will be modest.

Even this may be too optimistic, however. As our example demonstrates, intermediate structures often violate some principle or other. Therefore, successful interleaving of grammatical principles with parsing operations requires the existence of auxiliary procedures that determine whether a given partial structure can be presented to a given principle. Fong (1991) shows that the computational load that results from the application of these procedures in many cases outweighs the benefits of interleaving. He concludes, in the context of a GB-based parsing system, "[...] that parsing times will increase dramatically as more principles are interleaved". The key problem is that the auxiliary procedures do not only apply to partial structures that ultimately give rise to a successful parse, but also to those that eventually fail. This point suggests that interleaving is not a viable alternative to compilation. The number of failing structures that must be considered is larger if the structure builder is less restricted. The structure builder is less restricted if fewer principles have been compiled.

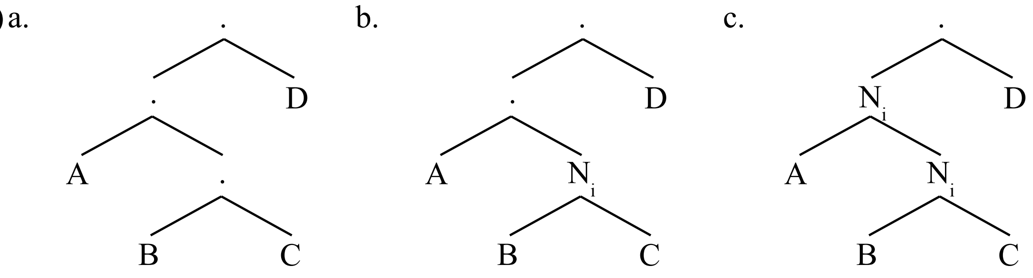
## 6 The benefits of underspecification

As just discussed, Inclusiveness – if conceived of as a filter – can only apply to a given structure if either the final input symbol is reached or the structure is placed on a left branch. This allows the parser to generate too many candidate structures before the principle can have its filtering effect.

Intuitively, a parser that allows underspecification of node content might be able to circumvent this problem. The idea would be that, rather than building many fully specified trees, the parser initially builds only a single underspecified one. Once enough information is available, node content is fixed in an efficient manner. Such a strategy can contribute to computational efficiency, but does not constitute an alternative to compilation.

To see what the potential benefits of underspecification amount to, consider a uniformly right-branching structure that is deemed complete and has been put on a left branch, as in (16a). At this point, there are $c$ choices for the content of the node directly dominating $B$ and $C$. Assuming an interleaved application of Inclusiveness, one of these will be selected (let us say $N_i$; $N_i = C$ if the grammar is right-headed). Next, there are $c$ choices for the node that immediately dominates $A$, of which again one will survive application of Inclusiveness. Therefore, the overall number of trees that the parser presents to Inclusiveness for a right-branching structure dominating $n$ nodes will never exceed $n \cdot c$, which seems acceptable.

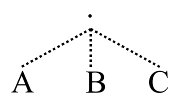
(16)



The overall picture is, however, less rosy than this narrative suggests. The essence of the underspecification strategy is to avoid commitment to any particular analysis. To begin with, there is abundant evidence that language computation has a high degree of incrementality, probably in order to associate the input with an interpretation as early as possible. This is apparent from garden path phenomena, amongst other things.

Moreover, the discussion of (16) presupposed commitment to tree shape in the absence of commitment to node content. But it is hard to see on what basis a parser could decide on a particular tree shape if all nonterminal nodes in the tree lack features. For example, the constituent containing *A*, *B*, and *C* may be placed on a left branch because it is selected by *D*. However, that selectional relation requires that the relevant constituent have selectable properties. In other words, its root node cannot remain underspecified. Much the same is true of the internal structure of the constituent under discussion. In other words, if the parser truly fails to commit to node content, it must also fail to commit to tree shape. This implies that, when *D* is encountered, the state of the parse is that in (17); that is, an unanalyzed string.

(17)



A cursory inspection of the literature suffices to establish that those researchers who rely on underspecification do indeed also make use of compilation in order to avoid postponement of commitment. For example, Marcus 1987, Marcus et al. 1983, Berwick and Weinberg 1985, and Gorrel 1995 all compile at least X'-theory (the precursor of Inclusiveness) and lexical selection (θ-theory and c-selection).

## 7 The status of the grammar

The discussion so far suggests that a description of the language faculty at the computational level should acknowledge two modules, namely a generator (or encoder) and a parser (or decoder), each employing a compiled special-purpose rule system. This implies a very much reduced role for a separate module of grammar at this level of description: it would be largely redundant. We thus arrive at the model in (6), in which the grammar is a description of the language faculty that abstracts away from computation, a description at what we called the cognitive level. This model allows a reconciliation of results in theoretical and computational linguistics. Grammatical theory is valid as a description of the language code, while compiled grammars are part of the encoding and decoding algorithms.

Since both the generator and the parser contain a rule system that instantiates the grammar, one may wonder what the point is of claiming mental reality for grammatical principles in addition to this? Indeed some linguists believe that there is no need to introduce a separate level of description for the grammar. The maxim that the grammar *is* the parser (endorsed by Phillips 1996, 2003, Kempson et al. 2001, and Steedman 2000, amongst others) is indicative of a research program along these lines. We are sympathetic to this kind of work insofar as it is aimed at developing a model of the language faculty at the computational level. After all, it directly addresses the concerns expressed above. We believe, however, that the principles of competence grammar have a right of existence independently from their implementation in parser and generator.

What is our justification for this claim? In general, a level of description is justified if it captures generalizations that cannot be captured at alternative levels of description. Marr gives as an example the case of some gas in a bottle. If one wants to describe the thermodynamic properties of such a mini system (temperature, pressure and volume), and the relationships between these properties, one does not use a large set of equations that each describe the movements of one of the particles involved. Rather we use a higher level of description to answer questions that could otherwise not be addressed. This is not just for practical reasons. The laws of thermodynamics are statistical in nature and, as a matter of logic, statistical statements cannot be derived from non-statistical premises, unless a specific theory linking them is formulated (see Popper 1992 for extensive discussion).

In fact, it is fair to say that in certain complex systems the organization of lower levels of description is explained by properties that only reveal themselves at higher levels of description. A simple example is that of human artifacts. These always have a function that, at least to some extent, explains their design and hence their physical realization. But functions cannot easily be described at the level of physics. Similar considerations extend to evolutionary development, where one can talk about the function of an organ, its design (given that function), and its physical instantiation (given that design).

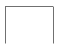
Such top-down explanations are never the whole story, as characteristics of higher levels can often be realized in different ways at lower levels. In the case of artifacts there might be various designs that realize a function and various physical realizations of a design. Which choices are made depends on various extraneous considerations, such as fashion, and the availability and price of certain materials; other relevant factors may be given by limitations at lower levels of description, such as the laws of physics. In the case of evolution the direction of development is partly determined by properties of the organism at earlier stages, as well as by properties of the environment. This notwithstanding we feel that any attempt at explaining artifacts or evolution must partly be top-down, if only because this allows us to distinguish those aspects of physical realization that are crucial from those that are coincidental. An exclusively bottom-up (or reductionist) approach must fail in this respect.

Before we work out a more detailed motivation for the cognitive level in (6), we clarify what is at stake by considering another phenomenon that requires multiple levels of description, namely cipher-based communication. We choose this particular example, because it shares a number of properties with communication through language.

## 8 The skytale cipher

A cipher relates a readable text (plaintext) to an encrypted text (ciphertext), and can minimally be characterized at two levels, namely as a mathematical function from plaintexts to ciphertexts and as a procedure that performs this mapping. The example we consider here is a system of encryption developed by the Spartans in the fifth century BC. The cryptographic device used is a skytale, a cylinder around which a strip of parchment is wound. Encryption takes place by writing the plaintext message along the length of the cylinder and unwinding the parchment strip. Decryption requires the strip to be wound around a cylinder of identical diameter, after which the plaintext message will reappear. The figure in (18) illustrates the encoding of a short message using a skytale that allows four lines of text and a length of parchment strip that allows five characters per line. (For ease of exposition we mark empty slots with a dash.)

(18)



|  |  |  |  |  |
|---|---|---|---|---|
| T | H | I | S | I |
| S | T | H | E | P |
| L | A | I | N | T |
| E | X | T | - | - |
|  |  |  |  |  |

When the strip is unwound, the text is no longer intelligible (we have rotated the letters to increase legibility):

(19)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  | T | S | L | E | H | T | A | X | I | H | I | T | S | E | N | - | I | P | T | - |  |

More abstractly, we can say that the plaintext in (20a) has been encoded as the ciphertext in (20b).

(20) a. THISISTHEPLAINTEXT--
b. *TSLEHTAXIHITSEN-IPT-*

The skytale cipher is one of transposition: it scrambles the letters that make up the plaintext, but does not change them. The scrambling algorithm divides the plaintext into columns and then places each column at the bottom of the one that precedes it. We can describe this algorithm as a function that takes as its input the position of a character in the plaintext (say *x*) and that returns the position of that character in the ciphertext (say *y*). What transposition takes place depends on the length of the parchment strip and the diameter of the skytale, as these determine the number of lines ($L_{max}$) and the number of characters per line ($C_{max}$). The type of transpositions that can be implemented by the procedure sketched above are given by the formula in (21), where *L* stands for a line of text in the encoding procedure (for (18) the value of *L* varies from 1 to 4).

(21) $(L - 1) \cdot C_{max} + 1 \leq x \leq L \cdot C_{max} \mid y = L_{max} \cdot (x - ((L - 1) \cdot C_{max} + 1)) + L$

In order to understand this formula, consider how the position of a character in (18) is related to its position in (19). There are two factors that are relevant. First, the column that contains the character determines the extent to which the character is shifted to obtain its ciphertext position. Second, the character's position in its column remains unchanged. The first factor is expressed by $L_{max} \cdot (x - ((L - 1) \cdot C_{max} + 1))$, while the addition of *L* expresses the constancy of positions internally to 'shifted' columns.

In the case at hand, $L_{max}$ equals 4 and $C_{max}$ equals 5. This implies that we can derive from the formula in (21) four more specific formulas, each corresponding to one line. (22a) states that if a character in the plaintext occupies one of the positions 1, 2, 3, 4 or 5, its position in the ciphertext will be as defined by the equation given. (22b-d) give equations for characters whose positions are 6 through 10, 11 through 15, and 16 through 20, respectively.

(22) a. $1 \leq x \leq 5 \mid y = 4 \cdot (x - 1) + 1$
b. $6 \leq x \leq 10 \mid y = 4 \cdot (x - 6) + 2$
c. $11 \leq x \leq 15 \mid y = 4 \cdot (x - 11) + 3$
d. $16 \leq x \leq 20 \mid y = 4 \cdot (x - 16) + 4$

If the formulas in (22) are applied to positions 1 through 20 in the plaintext, they are mapped to positions in the ciphertext as indicated by the table in (23). This mapping correctly characterizes the transposition in (20).

(23)

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 1 | 5 | 9 | 13 | 17 | 2 | 6 | 10 | 14 | 18 | 3 | 7 | 11 | 15 | 19 | 4 | 8 | 12 | 16 | 20 |

Several aspects of the skytale cipher are relevant to our discussion. To begin with, it is clear that one should distinguish between the mathematical properties of the transposition as given in (21) and (22) and the procedures of encoding and decoding. Suppose that a Martian scientist is presented with a number of plaintexts and their corresponding ciphertexts in the format in (20). He may well discover the cyclical nature of the cipher and come to understand that the key to any cipher of this type consists of two parameters (namely, $L_{max}$ and $C_{max}$). He may subsequently publish a paper in which he proposes the formulas in (21) and (22) and claim that these reflect the reality of the encryption method.

All this would be entirely reasonable, but it is clear from the example at hand that understanding the mathematical properties of the cipher does not identify the actual method of encryption. It merely constrains the class of candidate encoding devices.

It is also clear, when we consider the Spartans who actually used this cipher, that knowing how to use the skytale and the strip of parchment allows one to encode and decode without any knowledge of the formulas in (21) and (22). It is enough that these formulas are 'built into' the encryption procedure. The fairest description of the situation is to say that the mathematical properties of the cipher emerge from the encryption and decryption procedures.

The analysis of the skytale cipher exhibits properties typical of information-processing systems. First, the two descriptions to which the cipher yields are equivalent in that both relate the plaintext in (20a) to the ciphertext in (20b). Second, we can say that different algorithms may implement the same cipher. Suppose, for example, that one encrypts the message in (20a) using a table like (23) rather than a skytale and a strip of parchment. We want to be able to say that this procedure instantiates the same cipher as the one used by the Spartans, but such a generalization can only be made at the level of mathematical description. In other words, the higher level of description abstracts away from certain properties of the lower level. Third, the two levels of description are optimized for their respective functions. While the mathematical description captures the regularities that are the object of study for our Martian scientist, skytale and parchment strip facilitate easy encryption and decryption. Optimization at each level of description results in lack of transparency between these levels. (There is nothing like a mathematical function in the manipulation of stick and strip, or vice versa.)

Consider how the Martian scientist could defend himself against criticism that the mathematical description of the cipher is dispensable, given that it is implied by the methods of encryption and decryption. To begin with, as just mentioned, the mathematical description captures a generalization that cannot be stated at the algorithmic level, because there is more than one algorithm that can compute the same cipher. Furthermore, any communication through a cipher must necessarily yield to a mathematical description of the type in (21) and (22). This is because successful communication relies on the existence of regularities in the association of plaintext and ciphertext and these regularities constitute a description of the cipher at the mathematical level. Finally, the mathematical properties of ciphers are crucial to their selection by code makers. Ciphers are selected on the basis of two factors: they must be difficult to break, but easy to use. The former factor is largely determined by the nature of the mathematical function that describes them, while the latter is a matter of the usability of the associated encryption and decryption devices.

In the case of the skytale cipher, it seems unlikely that its mathematical properties were selected by its users and determined the nature of the encryption procedure. The design of modern ciphers, however, is partly driven by selection for their mathematical properties, which determine, among other things, their breakability. A recent example of selection for higher level properties of a cipher concerns the wish to avoid distribution of keys prior to cipher-based communication. This can be achieved by developing an asymmetric cipher; that is, a cipher which does not use the same key for encryption and decryption. This asymmetry would allow the prospective recipient to make the encryption key public, while not disclosing the decryption key to anyone. Subsequently, the sender can use the public key to encrypt a message that can only be decrypted by the holder of the private key.

Once the idea of an asymmetric cipher was hit upon, the race was on to find a mathematical function that could make it a reality (see Singh 1999 for a vivid description). It will be clear that no one was concerned at this point with the instantiation of this function. In other words, an explanation of the design of the associated encryption and decryption procedures must be partially top-down.

## 9 Multiple levels of description in language

In the previous section we have clarified why multiple levels of description are required for an information-processing system like a cipher. It is striking that the three properties we have attributed to the skytale cipher also hold of the model in (6).

First, the mapping expressed by the grammar at the cognitive level is logically equivalent to the computations carried out by the performance systems.

Second, the cognitive level is more abstract than the computational and neurological levels. As a result, it generalizes across parser and generator, which are unlikely to employ the same procedures, given their vastly different tasks. This point can be reinforced by a thought experiment. Suppose we build two computers that have radically different underlying architectures that, moreover, employ radically different operating systems. Suppose furthermore that both machines pass a kind of Turing test for language: their performance is indistinguishable from that of native speakers of some natural language. In this situation one would want to say that both computers speak that language, but the notion of identity required for this is unstateable at the hardware level or in terms of the operating system. The relevant notion of identity is based on regularities in the mapping between sound and meaning. The function that specifies these regularities is the grammar.

Third, descriptions at the cognitive and computational level are optimized with respect to their function. The grammar makes explicit regularities in the mapping between sound and meaning in a non-redundant manner, while the performance systems operate with a set of locally applicable rules that may well be highly redundant (such as the sets of rewrite rules referred to in section 4). This optimization results in a non-transparent relation between grammatical principles and the performance mechanisms from which they emerge.

This lack of transparency between the cognitive level and lower levels of description provides a solution to what one might call the paradox of perfection. As Chomsky and others have pointed out, biological systems are typically redundant and robust, while the various components of the grammar have shown to lend themselves to non-redundant analyses (see for example Chomsky and Lasnik 1993:515 and Brody 1998, 2003). Does this pose a problem for the view that grammar is a biological system? We do not think so. The relevant notion of biological system corresponds to a system at a lower level of description. Indeed, as just mentioned, the performance systems are likely to be highly redundant; the same is likely to be true of their physical realization at the neurological level. But there is no reason why the grammar, which describes a biological system at a higher level of abstraction, should not be perfect or elegant.

This solution to the paradox of perfection shares certain properties with that proposed by Li (1997), who argues that a functional description of a system might be perfect, even if the underlying biological reality is not. Li suggests that our study of grammar is essentially the study of a black box. His claim is that the grammar has the property of perfection because the rules that linguists formulate to relate the input and output of the black box are perfect. On this view, further discoveries about its internal workings may therefore reveal that the proposed rules are an inaccurate or even incorrect description of the underlying biological reality. Clearly, Li's interpretation of grammatical theory is instrumentalist: it is simply a convenient way of talking about the language faculty but does not necessarily capture anything real. (On our reading, Culicover and Nowak (2003) adopt a very similar position.)

There are good, and by now familiar, arguments against an instrumentalist interpretation of grammar. For a start, the thought experiment described above demonstrates that the grammatical description of a language captures a notion that cannot be stated at the algorithmic level. Hence, a growing insight into the biological reality that underlies language cannot replace grammatical theory. (It cannot falsify it either, unless accompanied by an explicit theory about the way in which grammatical properties emerge from the biological level.)

Furthermore, the logic of vocal communication dictates that sound and meaning must be associated in a regular manner and hence yield to a grammatical description. There simply cannot be successful communication in the absence of a shared code. This conclusion holds irrespective of the way this code is computed by speaker and hearer.

Finally, it can be argued that the principles of grammar capture abstract properties of human language that have been shaped by evolutionary pressures. If true, this implies that natural selection is for genes that express themselves in neurological structures that carry out computations that can implement particular (adaptive) grammatical principles. But if natural selection ultimately targets grammatical principles, then a description of human language at the level of grammar is indispensable. The role of evolution in the design of language is comparable to that of code makers in the development of ciphers. We elaborate on this argument in the next section.

## 10 Grammar and evolution

What would it take to demonstrate that principles of grammar are shaped by natural selection, and hence not merely instruments that linguists use to talk about the language faculty? An argument to that effect must begin by establishing that grammar is a complex trait. A simplex trait could be the result of an accidental mutation, but a complex trait must have been shaped by natural selection.

The position that language is indeed a complex trait has been forcefully defended by Pinker and Bloom (1990), Jackendoff (1992, 1994, 2002) and Pinker (1994, 2003). By contrast, Berwick (1998) and Hauser, Chomsky and Fitch (2002) argue it is an extremely simple combinatory mechanism, inserted as a bridge between pre-existing conceptual abilities and the systems of vocalization. Pinker and Jackendoff (2004) present a detailed critique of Hauser, Chomsky and Fitch's paper, giving empirical arguments that language is special in many more ways than in exhibiting a recursive syntax. We agree with their assessment (see also note 4).

Chomsky (2004) disputes the claim that language is a complex trait on the basis of the archaeological record. About fifty thousand years ago, there seems to have been a sudden cultural spurt, evidenced, among other things, by cave paintings. If this development was triggered by the emergence of human language, there would simply not have been enough time for a complex trait to develop.

The premise on which this argument is based is not itself backed up by evidence. Although cultural developments may be dependent on language, it is not clear that language is a sufficient condition for them. Furthermore, there *is* archaeological evidence that the evolution of language may have begun much earlier. Martinez et al. (2004) have analyzed the bones of the middle ear belonging to *Homo Heidelbergensis* in order to estimate the range of sound frequencies that they were most sensitive to. A study of five skulls dating back 400,000 years revealed that their owners would have been highly attuned to sounds between three and five kHz. This is remarkably similar to the sensitivity of modern humans, but very different from that of Chimpanzees, our nearest living relatives. It is generally assumed that the sensitivity range in modern humans is due to a specialization for understanding the spoken word. Since *Homo Heidelbergensis* was not one of our direct ancestors, this discovery would trace back the development of language to a shared ancestor who lived about 500,000 years ago. We do not want to suggest that this argument is conclusive, but it does make it clear that the archaeological record is open to more than one interpretation.

The second step in our argument must be to show that principles of grammar contribute to what is adaptive about language. This might seem difficult, as there is disagreement about the primary evolutionary advantage of language. Some researchers claim that it evolved to reap the benefits of communication (see Bickerton 1990, Nowak and Komarova 2001, Pinker 1994, 2003, and Pinker and Bloom 1990), while others support the view that the use of language for communication is secondary and its initial benefit was to allow thinking (conceived of as talking to oneself). Chomsky (2004) is one advocate of this position; he also mentions Jacob 1982 and Tattersall 1998 in this context. However, the two evolutionary scenarios have in common that natural selection will favour a language faculty that can express more propositional content over one with less expressive power, provided linguistic representations are sufficiently unambiguous to facilitate explicit thought or communication. For example, the ability to encode grammatical dependencies such as θ-role assignment and binding is adaptive in principle, because it makes it possible to achieve interpretational effects that cannot be achieved otherwise. However, if these dependencies are insufficiently constrained, a single structure can be associated with a multitude of interpretations, which would undermine their usefulness. In the extreme case, where any word can be related to any other word in an utterance, the beneficial effects of dependencies are lost altogether. In sum, what is favoured is a language faculty that allows many different grammatical dependencies, as long as these are sufficiently constrained.[4]

As we have argued, the principles that constrain grammatical dependencies (such as Inclusiveness) cannot be transparently realized in the performance systems (for example, as filters). Therefore, the evolutionary pressure described above targets more abstract properties of the language faculty: the grammar described at the cognitive level. If this argument is correct, an instrumentalist interpretation of grammatical theory is untenable.

## 11 Language acquisition

In the final three sections of this paper we confront our model with some arguments for a grammar module at the same level of description as the performance systems. Perhaps the best-known argument for this position has to do with the complexities of language acquisition. It is often claimed that successful acquisition relies on the setting of a limited number of (usually binary) parameters that together with various universal principles define the space of possible languages.

This view of acquisition is potentially threatened by our view that UG is a description of the language faculty at the cognitive level. If this is true and there is no transparent mapping between cognition and computation, parameters may only be available to the linguist, and not to the language learner. However, in the same way that principles may be compiled into the parser (and generator), compilation of parameters is at least a logical possibility. In fact recent work in parameter theory suggests that compilation is necessary, as we now explain.

Gibson and Wexler (1994) argue convincingly that off-line reasoning about the grammar cannot be part of the process of language acquisition. For this reason, these authors propose a learning strategy according to which a child who fails to parse the current input randomly

[4] Note that this line of argumentation also challenges the view that language is a simplex trait. The 'insertion' of a combinatory operation will not by itself yield evolutionary benefits, because the ability to express propositional content is adaptive only if it is expressed relatively unambiguously. This requires that any recursive operation co-evolves with a set of constraints, which implies that language is a complex trait.

changes one parameter value and subsequently attempts a reparse. If successful, the new parameter setting is adopted. Otherwise, the initial settings are retained.

In various publications, Janet Dean Fodor and William Sakas (Fodor 1998a,b, Sakas 2000, and Sakas and Fodor 2001) have argued that a significant improvement over this strategy is possible if its randomness can be eliminated. This is possible if off-line inferences about the consequences of parameter setting are built into the acquisition model. More specifically, these authors suggest that universal grammar makes available a set of treelets (or partial trees) that are used in parsing. Parameter setting is conceived of as the selection from this universal set of those treelets that are appropriate to the target language. The child's learning strategy upon encountering an unparsable input, given its current grammar (a language-specific set of treelets), is to select any additional treelets from the universal set that enable it to complete the current parse. These additional treelets are added to the language-specific set if the input is parametrically unambiguous (that is, there are no alternative treelets in the universal set that allow a successful parse). This model of language learning has been shown to be superior to more traditional theories.

Fodor and Sakas's proposal fits in very well with the theory of the competence-performance distinction defended here: saying that universal grammar takes the form of a set of treelets in acquisition and parsing is tantamount to admitting compilation of grammatical principles. This is because the treelets will not only encode parameters, but must each also adhere to the well-formedness constraints imposed by universal grammar. (In fact, if all treelets are taken to consist of a mother node and two daughters, the proposal is a notational variant of compilation through rewrite rules.)

We believe a stronger conclusion is warranted: Fodor and Sakas's proposal will require some structuring of the universal set of treelets. The point of parameter theory is to structure the learner's search space in such a way that it is easier to converge on the correct grammar. There are two ways in which parameter setting does so. To begin with, setting a parameter does not only imply that a pattern is added to the developing language, but also that alternative patterns are excluded. Furthermore, it might be that the acceptability of more than one pattern depends on a single parameter, so that adding one pattern to the developing language implies adding several other patterns as well (this is the idea behind so-called macro-parameters).

These effects must of course be captured. Even for parameters whose structural effects are quite local and which hence can be successfully expressed as the presence or absence of a particular treelet, it must be assumed that selection of one treelet precludes selection of an alternative. For example, if a treelet is chosen in which the base position of the object follows the verb, then the alternative treelet in which the object precedes it is no longer selectable. If this restriction is not enforced, we cannot truthfully say that we have implemented the OV/VO-parameter.

Even more structure in the set of treelets must be postulated if parameters have non-local effects. Suppose that the OV/VO-parameter does not only govern the position of the object with respect to the verb but also the position of other material (such as particles, certain adverbs, resultative secondary predicates, etc.). In that case, the selection of a treelet for one of these constructions must entail automatic selection of other treelets.

We conclude that Fodor and Sakas's proposal, if correct, will require a nontransparent compilation of parameter theory in much the same way that parsing requires a nontransparent compilation of the grammar.[5] Just as the abstract nature of linguistic principles interferes with

[5] One might think that it is not necessary to structure the set of treelets in this way, because the regularities in question are already present in the data that the child is exposed to. It is true that a computer program selecting treelets on the basis of a natural language database would converge on the correct set of treelets, even if the set made available by universal grammar was unstructured. However, if one asks why language displays the regularities that it does, it would beg the question to attribute them to the nature of the data.

parsing efficiency, the abstract nature of parameters leads to an increase in the computational burden associated with language acquisition. In both cases, the solution seems to lie in an optimization of the language faculty for the computational task at hand and restricting the scope of universal grammar to a more abstract level of description.

## 12 Aphasia and the shared languages of perception and production

Two further arguments for a grammar module at the same level of description as the performance systems can be found in the following continuation of the quote on p.2 from Chomsky (2000):

> The language faculty has an input receptive system and an output production system, but more than that; no one speaks only Japanese and understands only Swahili. These performance systems access a common body of information, which links them and provides them with instructions of some kind. The performance systems can be selectively impaired, while the cognitive system remains intact […].

We believe that these conclusions are open to criticism.

To begin with, we doubt that there is a pattern of aphasia that conclusively shows that the grammar must be a knowledge module consulted by parser and generator. There are three main cases to consider. First, a person might have lost the ability to produce language, having retained normal perception. On Chomsky's view this would imply that the output production system has been damaged, while the grammar and the input receptive system are unaffected. Second, a person might have lost the ability to perceive language, having retained normal production, implying damage to the input receptive systems, with the grammar and output production system functioning normally. Finally, a person might have lost both language production and perception. This could be indicative of a failure of the grammar, damage to the two performance systems, or both. But these situations can be described equally well if the grammar is not a module at the computational level: the first pattern of aphasia corresponds to a breakdown of the generator, the second to a breakdown of the parser, and the third to a breakdown of both.

The second point in the quote above concerns the question why the languages of perception and production are systematically identical. Chomsky's answer is that production and comprehension make use of the same knowledge base. But this does not solve the puzzle. After all, if it were adaptive for the languages of comprehension and production to be different, evolution could very well have led to separate grammars for the two systems.

In fact, there are obvious arguments suggesting evolutionary pressure towards a common language for perception and production, whether language is conceived as a tool for communication or for thought. This is compatible with the view that parser and generator share a knowledge base, but it does not force us to adopt this position. For the sake of argument, let us assume that the grammar employed in perception is constant across individuals. The perception system is a crucial component of the evolutionary environment to which the production system is sensitive. Hence, the production system must have evolved to produce utterances that optimally fit the perception system. The same line of argumentation implies that the perception system must have evolved in such a way that it optimally processes the output of the production system. Therefore, evolutionary negotiation will result in parser and generator settling on the same code.

Of course, there is variation between languages and this means that the above argument merely guarantees that parser and generator adhere to the same universal grammar. However, the same evolutionary pressures favour the development of mechanisms that ensure identical parameter setting in parser and generator. There is a very likely candidate for such a mechanism, namely feedback in production: hearing oneself speak. There is evidence that monitoring one's speech is extremely important in production. A subject wearing headphones that delay this auditory feedback by 200 msec. or so displays severely disrupted speech. He or she will sometimes even produce uncontrollable stuttering (see Jackendoff 1987 and references mentioned there). In light of this, it is suggestive that language perception systematically precedes language production in children.

## 13 Grammaticality judgements

A final argument for the existence of a competence module concerns the distinction that native speakers make between sentences that are hard to process and those that are ungrammatical. That speakers can make this distinction is all the more remarkable in view of the fact that giving a grammaticality judgement is a matter of performance. In order to know that a mistake has been made, there has to be some yardstick against which performance is judged and the usual claim is that this is the role that the competence module plays.

Given the proposal we have put forward, the question arises how we could characterize the relevant contrast. Of course, we accept that there is a difference between ungrammatical sentences and sentences that are hard to parse. We also accept that native speakers have a certain degree of access to this distinction. What we do not accept is that this distinction must be grounded in a competence module at the computational level. Recall that we have taken the parser and generator to contain a compiled grammar. Hence, the performance systems will not be able to assign a complete structure to strings that do not adhere to the hearer's grammatical code. This situation must be distinguished from one in which it is hard – but not impossible – to find a complete structure for the input string, such as in the case of garden path sentences. Native speakers can make the distinction between these two cases by producing the string in question several times and using it as input to the parser through the feedback mechanism discussed above. This 'looping' may reveal that a structure is in fact available although it was not found on the first attempt.

As is well-known, a native speaker may still assign an interpretation to an input for which no complete structure is found. In particular, if the input string can be parsed into several unconnected structures, discourse-level processes may be able to integrate the semantics of these fragments into a coherent proposition. (It is probably not an overstatement that this type of processing is very common in normal conversations.) For certain ill-formed sentences it is also conceivable that the performance mechanisms carry out repairs that reconcile the input with the grammatical code. A low-level example of such a repair is phoneme restoration (Warren 1970), but there might be higher-level repairs that locally affect phoneme, morpheme or constituent order. In this respect, there is no difference between our proposal and the standard view.

One aspect in which our proposal does diverge from the standard view is that it has something to say about the well-formedness of incomplete structures. The competence grammar, conceived of as a filtering device, will rule out all incomplete structures. However, such structures are frequently produced in discourse and not perceived as deviant, beyond the fact that they are incomplete. Someone listening to the monologue in (24), for example, will not experience the various incomplete utterances as ill-formed.

(24) I said… I mean… Susan's unlikely to do something like that. What d'you think she… What would you say if she'd really done that?

By contrast, irrespective of the context in which it is uttered, (25) is perceived as not just incomplete but deviant (see Jackendoff 1987:106).

(25) What movies do you know lots of people who…

Our view of the performance mechanisms is that they are designed to operate in accordance with a grammatical code. However, as opposed to the competence grammar, such mechanisms must be able to deal with incomplete structures, as intermediate stages in the processes of parsing and generation are by necessity incomplete.[6] To put the same thing differently, in normal circumstances the performance mechanisms operate in accordance with the grammatical code and hence any partial structures they assign are reflections of that code. This implies that the judgement associated with (25) signals that the perceptual mechanisms cannot find a single structure that covers the entire string. Since no such failure occurs in any of the incomplete structures in (24), these are not considered deviant.

Standard grammaticality judgement tasks always involve judgements of strings that are taken to form complete sentences. Therefore, the task comes with the instruction to check whether a single complete structure can be assigned to the input. While such a task can reveal a great deal about the language faculty, the fact that it targets an idealized language means that what is acceptable in conversation will fail in the grammaticality judgement task: speakers will judge both the incomplete WH questions in (24) and (25) ungrammatical.

The data in (24) and (25) are not completely beyond the reach of theories that assume a grammatical module (conceived of as a filtering device), provided that a further module is added to the language faculty whose task it is to uncover possible continuations of incomplete structures. The resulting completed structures can be presented to the grammar in normal conversation, but the sentence completion module would have to be switched off in grammaticality judgement tasks. This will work, but it seems ad hoc.

## References

Abney, Steven (1988). On the Notions 'GB Parser' and 'Psychological Reality'. In Steven Abney (ed.). *MIT Parsing Volume, 1987-88*. Center for Cognitive Science, MIT.

Abney, Steven (1989). A Computational Model of Human Parsing. *Journal of Psycholinguistic Research* 18: 129-144.

Abney, Steven (1991). Parsing by Chunks. In Robert Berwick, Steven Abney and Carol Tenny (eds.) *Principle-Based Parsing: Computation and Psycholinguistics*, 257-278. Dordrecht: Kluwer Academic Publishers.

Berwick, Robert (1985). *The Acquisition of Syntactic Knowledge*. Cambridge: MIT Press.

Berwick, Robert (1998). Language Evolution and the Minimalist Program: The Origins of Syntax. In James Hurford et al. (eds.) *Approaches to the Evolution of Language: Social and Cognitive Bases*, pp. 320-340. Cambridge: CUP.

Berwick, Robert, Steven Abney and Carol Tenny (eds.) (1991). *Principle-Based Parsing: Computation and Psycholinguistics*. Dordrecht: Kluwer Academic Publishers.

Berwick, Robert, and Amy Weinberg (1984). *The Grammatical Basis of Linguistic Performance*. Cambridge: MIT Press.

Berwick, Robert, and Amy Weinberg (1985). Deterministic Parsing and Linguistic Explanation. *Language and Cognitive Processes* 1: 109-134.

Bickerton, Derek (1990). *Language and Species*. Chicago: University of Chicago Press.

Brody, Michael (1998). The Minimalist Program and a Perfect Syntax. *Mind and Language* 13: 205-21.

Brody, Michael (2003). *Towards an Elegant Syntax*. London: Routledge.

Chomsky, Noam (1965). *Aspects of the Theory of Syntax*. Cambridge: MIT Press.

Chomsky, Noam (1995a). Bare Phrase Structure. In Gert Webelhuth (ed.) *Government and Binding Theory and the Minimalist Program*, 383-439. Oxford: Blackwell.

Chomsky, Noam (1995b). *The Minimalist Program*. Cambridge: MIT Press.

[6] See Ristad (1993:115-116) for discussion.

Chomsky, Noam (2000). *New Horizons in the Study of Language and Mind*. Cambridge: CUP.

Chomsky, Noam (2004). Three Factors in Language Design. Ms. MIT. To appear in *Linguistics Inquiry*.

Chomsky, Noam, and Howard Lasnik (1993). The Theory of principles and Parameters. In Joachim Jacobs et al. (eds.) *Syntax: An International Handbook of Contemporary Research*, 506-569. Berlin: Walter de Gruyter.

Culicover, Peter and Andrzej Nowak (2003). *Dynamical Grammar*. Oxford: OUP.

Dennett, Daniel (1992). *Consciousness Explained*. London: Penguin books.

Earley, Jay (1970). An Efficient Context-Free Parsing Algorithm. *Communications of the Association for Computing Machinery* 14: 453-460.

Fodor, Janet Dean (1998a). Unambiguous Triggers. *Linguistic Inquiry* 29: 1-36.

Fodor, Janet Dean (1998b). Parsing to Learn. *Journal of Psycholinguistic Research* 27: 339-374.

Fong, Sandiway (1991). The Computational Implementation of Principle-Based Parsers. In Berwick et al. (1991), 65-82.

Frazier, Lyn and Charles Clifton (1995). *Construal.* Cambridge: MIT Press.

Gibson, Edward and Ken Wexler (1994). Triggers. *Linguistic Inquiry* 25: 407-454.

Gorrell, Paul (1995). *Syntax and Parsing*. Cambridge: CUP.

Hauser, Marc, Noam Chomsky and Tecumseh Fitch (2002). The Faculty of Language: What is It, Who Has It, and How Did It Evolve? *Science* 298: 1569-1579.

Jackendoff, Ray (1987). *Consciousness and the Computational Mind*. Cambridge: MIT Press.

Jackendoff, Ray (1992). Languages of the Mind. Cambridge: MIT Press.

Jackendoff, Ray (1994). *Patterns in the Mind: Language and Human Nature.* New York: Basic Books.

Jackendoff, Ray (2002). *Foundations of Language: Brain, Meaning, Grammar and Evolution*. Oxford: OUP.

Jacob, François (1982). The Possible and the Actual. Washington: University of Washington Press.

Kempson, Ruth, Wilfried Meyer-Viol and Dov Gabbay (2001). *Dynamic Syntax: The Flow of Language Understanding*. Oxford: Blackwell.

Kolb, Hans-Peter and Craig Thiersch (1991). Levels and Empty Categories in a Principles and Parameters Approach to Parsing. In Hubert Haider and Klaus Netter (eds.) *Representation and Derivation in the Theory of Grammar*. Dordrecht: Kluwer Academic Publishers.

Levelt, Willem (1989). *Speaking: From Intention to Articulation*. Cambridge: MIT Press.

Li, Yafei (1997). An Optimized Universal Grammar and Biological Redundancies. *Linguistic Inquiry* 28: 170-178.

Marcus, Mitchell (1980). *A Theory of Syntactic Recognition for Natural Language*. Cambridge: MIT Press.

Marcus, Mitchell (1987). Deterministic Parsing and Description Theory. In Peter Whitelock, Harold Somers, Paul Bennett, Rod Johnson, and Mary McGee Wood (eds.) *Linguistic Theory and Computer Applications*, 69-112. New York: Academic Press.

Marcus, Mitchell, D. Hindle and M. Fleck (1983). D-Theory: Talking about Talking about Trees. *Association for Computational Linguistics* 21: 129-136.

Marr, David (1982) *Vision*. New York: W.H. Freeman.

Martínez, Ignacio et al. (2004). Auditory Capacities in Middle Pleistocene Humans from the Sierra de Atapuerca in Spain. *Proceedings of the National Academcy of Sciences of the United States of America* 101: 9976-9981.

Neeleman, Ad and Hans van de Koot. (2002). The Configurational Matrix. *Linguistic Inquiry* 33: 529-574.

Newmeyer, Frederick (2003). Grammar is Grammar and Usage is Usage. *Language* 79: 682-707.

Nowak, M. A. and N. L. Komarova (2001). Towards an Evolutionary Theory of Language. *Trends in Cognitive Sciences* 5(7): 288-295.

Phillips, Colin (1996). *Order and Structure*. PhD dissertation. MIT.

Phillips, Colin (2003). Order and Constituency. *Linguistic Inquiry* 34: 37-90.

Pinker, Steven (1994). *The Language Instinct.* New York: Harper Collins.

Pinker, Steven (2003). Language as an Adaptation to the Cognitive Niche. In Morton Christiansen and Simon Kirby (eds.), *Language Evolution: States of the Art*. New York: Oxford University Press.

Pinker, Steven and Paul Bloom (1990). Natural Language and Natural Selection. *Behavioral and Brain Sciences* 13: 707-784.

Pinker, Steven and Ray Jackendoff (2004). The Faculty of Language: What's Special About It? Ms. Harvard University and Brandeis University. To appear in *Cognition*.

Popper, Karl (1992). *Quantum Theory and the Schism in Physics: From the Postscript to the Logic of Scientific Discovery*. London: Routledge.

Pritchett, Bradley (1992). *Grammatical Competence and Parsing Performance*. Chicago: The University of Chicago Press.

Ristad, Eric (1993). *The Language Complexity Game*. Cambridge: MIT Press.

Sakas, William Gregory (2000). Modeling the Effect of Cross-Language Ambiguity on Human Syntax Acquisition. In *Proceedings of CoNLL-2000 and LLL-2000*, 61-66. Lisbon.

Sakas, William Gregory and Janet Dean Fodor (2001). The Structural Triggers Learner. In Stefano Bertolo (ed.) *Language Acquisition and Learnability*. Cambridge: CUP.

Singh, Simon (1999). *The Code Book: The Secret History of Codes and Code-Breaking*. London: Fourth Estate.

Steedman, Mark (2000). *The Syntactic Process*. Cambridge: MIT Press.

Tattersall, Ian (1998). *The Origin of the Human Capacity*. New York: American Museum of Natural History.

Van de Koot, Hans (1990). *An Essay on Grammar-Parser Relations*. Dordrecht: Foris.

Warren, Richard (1970). Perceptual Restoration of Missing Speech Sounds. *Science* 167: 392-393.

Younger, Daniel. (1967). Recognition and Parsing of Context-Free Languages in Time $n^3$. *Information and Control* 10: 189-208.