Three Ways to Link Merge with Hierarchical Concept-Combination

Chris Thornton

In the Minimalist Program, language competence is seen to stem from a fundamental ability to construct hierarchical structure, an operation dubbed 'Merge'. This raises the problem of how to view hierarchical concept-combination. This is a conceptual operation which also builds hierarchical structure. We can conceive of a garden that consists of a lawn and a flower-bed, for example, or a salad consisting of lettuce, fennel and rocket, or a crew consisting of a pilot and engineer. In such cases, concepts are put together in a way that makes one the accommodating element with respect to the others taken in combination. The accommodating element becomes the root of a hierarchical unit. Since this unit is itself a concept, the operation is inherently recursive. Does this mean the mind has two independent systems of hierarchical construction? Or is some form of integration more likely? Following a detailed examination of the operations involved, this paper shows there are three main ways in which Merge might be linked to hierarchical concept-combination. Also examined are the architectural implications that arise in each case.

Keywords: Merge; Minimalist Program; hierarchical concept combination

1. Introduction

Hauser et al. (2002) note that underlying language there must be a faculty that is "hierarchical, generative, recursive, and virtually limitless with respect to its scope of expression" (Hauser et al., 2002: 1569). By implication, the language system must have, at its heart, an operator which can relate multiple objects to a single object in the formation of a hierarchical unit. Recursive application of this operator must be what gives rise to structurally complex expressions. The Minimalist Program dubs this fundamental operator 'Merge'. Chomsky deduces its existence as follows:

> An elementary fact about the language faculty is that it is a system of discrete infinity. Any such system is based on a primitive operation that takes n objects already constructed, and constructs from them a new object: in the simplest case, the set of these n objects. Call that operation Merge. Either Merge or some equivalent is a minimal requirement. With Merge available, we instantly have an unbounded system of hierarchically structured expressions. (Chomsky, 2005: 11-12)

ISSN 1450–3417

Chomsky also notes that Merge has the potential to be applied to conceptual entities. Used this way, he observes, the operator would implement a compositional medium of thought with an unlimited generative capacity. As he says,

> Emergence of unbounded Merge at once provides a kind of language of thought, an internal system that makes use of conceptual atoms (perhaps pre-existent) to construct expressions of arbitrary richness and complexity. (Chomsky, 2007a: 16)

No other internal compositional apparatus is needed, Chomsky maintains, to explain the productivity of thought. While acknowledging it "is often argued that another independent language of thought must be postulated", he considers the arguments not to be compelling (Chomsky, 2007a: 16).

The view is shared by, among others, Hauser (2009) and Hinzen (2009, 2012). For Hinzen, Merge's capacity to implement a language of thought is further evidenced by the inherently grammatical nature of thought. Hinzen argues that a "science of grammar is or can be a science of human thought because it uncovers the principles and organization of thought, and it can do so because our mode of thought is uniquely grammatical" (Hinzen, 2012: 642). This leads to the conclusion that "insofar as our mode of thought is species-specific and needs to find an explanation, grammar is the most likely such explanation" (Hinzen, 2012: 646). On this view, generative grammar is not only the mediation of language, but also of thought: It "is the essential mechanism we need, with no additional and independent language of thought required" (Hinzen, 2012: 647).

Underlying this position is the assumption that grammar (or more fundamentally Merge) is the only generative system of hierarchical construction the mind possesses. Chomsky rejects the possibility of there being any hierarchically generative system other that Merge, commenting that "To date, I am not aware of any real examples of unbounded Merge apart from language" (Chomsky, 2007b: 20). Hinzen takes the same position in the case of grammar more generally, arguing that

> there is no known non-grammatical way in which meanings of that specific kind arise, or on what generative system they should be based. (Hinzen, 2012: 646)

This raises the question of how to view hierarchical concept-combination. This is a conceptual mechanism of the envisaged type. Concepts are readily put into hierarchical combinations, and the process of doing so features prominently in ordinary thought. In conceiving of a residence consisting of a house and garden, or a bouquet consisting of a rose, orchid and tulip, or a meal consisting of a burger and fries, we put concepts together in a way that makes one the accommodating element with respect to the others taken in combination. A hierarchical structure is produced in which the accommodating concept forms the root. Since what is obtained is itself a concept, the procedure is inherently recursive. The garden that is conceived as contributing to the make-up of a residence can, itself, be conceived as consisting of a lawn and flower-bed. Concepts have the potential to be assembled compositionally, and multi-level structures can be obtained in this way.

Hierarchical concept-combination (HCC) makes use of conceptual entities to construct compositional structures of arbitrary depth, then. This is akin to the capacity Chomsky attributes to Merge, for using conceptual atoms to construct "expressions of arbitrary richness and complexity" (Chomsky, 2007a: 16). Are Merge and HCC the same operator, then? HCC implements a language of thought that is inherently compositional. Chomsky notes that Merge does too. Are these two languages of thought really the same language? Can we say that Merge = HCC?

The answer is not entirely straightforward. If Merge and HCC are the same mechanism, each should be able to fulfil the functions of the other. In practice, the relationship is not quite symmetrical. HCC distinguishes between an accommodating concept, and concepts that are accommodated *by* it. Merge makes no such distinction. While HCC can implement Merge, Merge can only implement HCC on certain assumptions, then. Setting this reservation aside, the two mechanisms can be seen as functionally equivalent, on which basis HCC might implement Merge, or Merge might implement HCC. But this poses its own problems. If we assume the two mechanisms are implemented separately—one in the language system and one in the conceptual system—that would imply a wasteful duplication of mental resources. If we assume there is only one implementation, there is the problem of explaining how this could simultaneously serve the role of Merge in the production of language, and of HCC in the production of hierarchical conceptualizations.

What is argued below is that there are two ways in which these two mechanisms might be connected. The less radical scheme keeps the language system in its usual form, but assumes that one of the two mechanisms provides services to the other on a client-server basis. This raises the question of how the provision is accomplished: What changes are implied for the interface between the language and conceptual systems? The more radical arrangement is one which keeps the conceptual system in its usual form, but assumes that HCC sends hierarchical *structures* to the language system for translation into symbol sequences. This raises different questions: How the translation is accomplished is clearly one. Counting the arrangement in which the two operators work independently, there are then three possible ways in which HCC and Merge might be related.

The remainder of the paper sets out the analysis in detail. There are five main sections in all. The section immediately to follow (Section 2) looks at hierarchical concept-combination from a more formal perspective. A shorthand is introduced which allows constructions to be specified in the briefest possible way. This is then used to develop a number of examples showcasing the compositional generativity of the mechanism. Section 3 then looks at Merge and considers the ways it may be related to HCC. The latter part of this section focuses specifically on the case where HCC is considered to contribute hierarchical structures to the language system, the 'semantics-first' arrangement as it is dubbed below. Section 4 then draws the evidence together and considers what is implied. Finally, Section 5 offers some concluding comments.

Some technical points should be mentioned at the outset. Material dealing with conceptualization follows the approach of Murphy (2002) in avoiding use of special fonts in the naming of concepts. Where there is any ambiguity, the phrase 'the concept of X' is used to indicate that X is a concept name. Following normal practice in linguistics, the dot '.' is used as a connective in phrasal

concept names. The concept of an old man thus has the name 'old.man'. Some of the material dealing with language makes use of interlinear glosses. All the glosses presented can also be found in the online resource 'The World Atlas of Language Structures' (Dryer and Haspelmath, 2011). References to the present contents of this resource use the acronym 'WALS'. The internet location of the resource is `http://wals.info`.

2. Hierarchical Concept-Combination

The capacity that concepts have, to be put into hierarchical combinations, has not been previously studied in any depth. This may be because the operation is such a familiar and ubiquitous feature of thought. The opportunity to form a construction of this type exists whenever one concept has the capacity to accommodate one or more others, taken in combination. Examples are easily concocted; e.g., a family that consists of a mother and child, or a salad consisting of lettuce, spinach and cucumber, or a garden consisting of a lawn and a flower-bed. In each case, concepts are brought together in a way that makes one the accommodating element with respect to the others. The idea constructed is a hierarchical structure in which the accommodating concept provides the root. Hierarchical concept-combination is not to be confused with *generic* concept combination, however. This is a more diverse process that has been modeled in a range of ways (e.g. Hampton, 1991; Thagard, 1997; Rips, 1995; Wisniewski, 1997; Costello and Keane, 2001; Hampton, 1997, 2011).

Since an idea derived by hierarchical combination of concepts is itself a concept, the operation is inherently recursive. We can conceive of a residence that consists of a house and a garden, where the garden is conceived as consisting of a lawn and flower-bed. Or we might conceive of a meal that consists of a steak and salad, where the salad is envisaged as consisting of feta and fennel. In such cases, a structure of two levels is obtained. In principle, new levels can be added without limit. Hierarchical combination of concepts can give rise to multi-level structures in this way.

A structure assembled in this way is not a concept hierarchy in any of the usual senses, however. It is not, for example, a generalization hierarchy (a taxonomy or 'is-a' tree). In the concept of a family consisting of a mother and child, the concept of a family does not generalize the concept of a mother. It does not play the role of a hyponym. Neither is the structure a part-whole hierarchy (a meronomy or 'has-a' tree). In a meronomy, each hierarchical entity is defined to be the composite of the cited parts. There can be only one whole for any combination of parts, whereas there can be multiple hierarchical accommodations for the same combination. A mother and child might be conceived as forming, say, a vocal duo, or a darts team, as well as a family.[1] Another standard form of concept hierarchy was established by Gentner (1983). In her model, each hierarchical entity is a predicate which defines a concept in terms of lower-level attributes and/or predicates. This differs from the situation in hierarchical concept-combination, where each hierar-

[1] An alternative way to make the distinction is to say that a part-whole hierarchy defines the restricted case of hierarchical combination in which the accommodating concept is just the concept of a composite (i.e., a set).

chical entity specializes a concept: It is a realization of the concept *constituted* in a specific way. Structures built by hierarchical concept-combination are not predicate structures for this reason.

An interesting aspect of hierarchical concept-combination is its productivity. Concepts can only be combined hierarchically in particular ways. We can conceive of a family that consists of a mother and child. But the idea of a mother that consists of a family and child makes no sense. These three concepts cannot be put together in this way. Because of its meaning, the concept of a mother cannot accommodate this combination. It cannot be the root of a hierarchical structure in which the accommodated combination includes the concept of a family.

It is the semantic properties of concepts which define the hierarchical combinations that can be formed, then. Accordingly, a given set of concepts gives rise to a particular set of structures. This set has the potential to be empty. Imagine, for example, we have just the concepts mother, child and father. There is no concept within this set that can accommodate any combination of the others. The implied set of hierarchical combinations is thus empty. But say we start with the set: singer, guitarist, drummer, duo, band. Multiple hierarchical combinations are then legitimized. One is the concept of a duo consisting of a guitarist and a singer (a singer-guitarist duo). Another is the concept of a duo consisting of a guitarist and a drummer (a drummer-guitarist duo). Since the concept of a duo can accommodate any pair of individuals, and there are three such combinations, there are three duo concepts that can be constructed.

Taking the semantic capacities of the band concept into account, the set of structures expands further. Since a band can also consist of any combination of music-making individuals, there is a band-based counterpart for each of the duo concepts. But a band that consists of multiple duos can also be envisaged. There are some two-level structures to be acknowledged in result. A band might combine a singer-guitarist duo with a singer-drummer duo, for example. This is another idea with a hierarchical structure of two levels. At one level, the duo concept is the accommodating element; at the other, the band concept is.

Another interesting aspect of HCC is its unboundedness. Infinitely many constructions can potentially be derived. A hierarchical combination of existing concepts constructs a concept that is inherently new: it is a compositional construction built from existing concepts, which differs from all of them. This new concept may enable new hierarchical constructions to be assembled, giving rise to further concepts and constructions in an ongoing way. Given a sufficient initial endowment of concepts, HCC can, in this way, provide an infinitely productive language of thought. The minimal requirement for infinite productivity is that every construction provides a concept that facilitates at least one further construction. This guarantees there will be infinitely many.

As noted above, HCC has not previously been studied as a conceptual mechanism in its own right. All work on concepts acknowledges this medium to some degree, however. In general, concept construction is modeled as application of a constructive operator to a set of existing concepts, e.g., taking the conjunction of the concepts male and unmarried to define the concept of a bachelor (Murphy, 2002). But this is closely related to hierarchical concept-combination. Any construction which involves application of a constructive operator to certain operands can also

be viewed as an act of hierarchical concept-combination. The accommodated combination equates to the set of operands, while the accommodating concept equates to whatever idea is realized by the construction, with the operands abstracted away. Put another way, the accommodating concept is whatever concept is imposed on the operands by the operator. For example, constructing a conjunction of concepts can be seen as forming a hierarchical unit in which the concept of conjunction is the accommodating element, and the conjoined concepts make up the accommodated combination. What is obtained is an instance of the accommodating concept, conceived as constituted in a particular way. The study of concepts takes the possibilities of HCC into account to this extent (e.g. Laurence and Margolis, 1999; Carey, 2009).

2.1. HCC in Shorthand

To demonstrate the practical possibilities of hierarchical concept-combination, examples of some complexity will need to be set out. Since English descriptions of complex structures quickly become unreadable, it is helpful to introduce a shorthand at this point. The convention henceforth will be that a hierarchical structure in which concept X is conceived as accommodating some combination of concepts, will be denoted by enclosing all the concepts in square brackets with X placed first. Thus

[**X** Y Z]

is shorthand for a hierarchical combination in which concept X is the accommodating element with respect to Y and Z. The accommodating concept—the hierarchical root—is emboldened for emphasis. There can be any number of concepts in the accommodated combination, and they are not in any order. Embedding of structure is then dealt with by bracketing in the obvious way. The concept of an X encompassing[2] the combination of a Y and Z, where the Z itself encompasses the combination of B, D and E, has the shorthand

[**X** Y [**Z** B D E]]

The examples introduced above can all be expressed more succinctly using this approach. The concept of a family made up of a mother and child, for example, can be expressed as

[**family** mother child]

The concept of a residence encompassing a house and garden, where the garden itself encompasses a lawn and flower-bed can be expressed as

[**residence** house [**garden** lawn flower-bed]]

Any construction in the shorthand relies on knowledge of the cited concepts, and how they are named in English. The basic knowledge of English that allows 'a family consisting of a mother and child' to be correctly interpreted is also required to interpret [**family** mother child]. The knowledge that 'family' names the concept

[2] The terms 'accommodating' and 'encompassing' are used interchangeably in characterizing the root element of a hierarchical unit.

of a family, 'child' the concept of a child etc., is part of what gives a construction its meaning. The only difference between an English specification of a hierarchical concept-combination, and one expressed using the shorthand, is that in one case accommodation is designated by the phrase 'consisting of', whereas in the other it is designated by square-bracketing and constituent positioning.

If cited concept names are considered to be well-defined, then the shorthand can legitimately be viewed as a formal notation for concept construction by hierarchical combination. The assumption that these terms are well-defined might be made on the grounds that the have definitions in English dictionaries. In principle, one might try to eliminate this appeal to knowledge of English by providing a formal definition of the concepts in question. This is ruled out in practice, since a general way of defining concepts has never been established (Carey, 2009).

Provided concept names are well-defined, expressions in the shorthand remain semantically precise regardless of their complexity. Since the only construction used is conceptual accommodation, it makes no difference how many times this is applied. If a 1-level construction is semantically precise, so too is a n-level construction, where $n > 1$. This can be illustrated using a three-level form. Consider the idea of a meal consisting of a steak and a salad where the salad consists of lettuce, cucumber and a dressing, and the dressing consists of oil, vinegar and salt. This idea has a precisely defined meaning provided we possess the cited concepts, and correctly interpret the indicated accommodations. The shorthand

[**meal** steak [**salad** lettuce cucumber [**dressing** oil vinegar salt]]]

is semantically precise on the same conditions.

The shorthand conforms to the following Backus-Naur (BNF) specification:

⟨spec⟩ ::= ⟨concept-name⟩
⟨spec⟩ ::= "[" ⟨spec⟩ ⟨spec⟩$^+$ "]"

The non-terminal ⟨spec⟩ denotes the specification of a concept. This is defined to be either the name of a concept, or a square-bracketed sequence containing two or more ⟨spec⟩. Use of the '+' superscript allows that hierarchical combinations can have any number of encompassed elements. Without this annotation, the effect would be to enforce combinations with only one.[3]

Viewing the shorthand as a formal notation raises the question of how it should be classified. What type of notation is this exactly? The convention of placing the accommodating concept first in sequence seems indicative of a prefix or Polish notation—one in which the operator of a construction is placed before its arguments. But this classification is incorrect strictly speaking. The operation implied is hierarchical accommodation *by means of* the initially placed concept. It is the initial concept deployed as the accommodating element. The initial concept is not itself the constructive operator, and the shorthand is not a prefix notation for this reason.

The shorthand is also not in any sense a version of predicate logic. This is an important caveat, as predicate logic has often been used as a way of specifying hierarchical concepts (e.g. Gentner, 1983). A hierarchical construction in predicate

[3] Notice that since the encompassed elements in a construction have no order, it is always the case that [**X** Y Z] = [**X** Z Y].

logic does not describe a hierarchical concept-combination. The operation of logical predication is not the operation of conceptual accommodation. In one case, the result is a truth value, in the other, it is the specialization of a concept. (This is one reason for avoiding a predicate-style format in which the accommodated combination is enclosed in round brackets, and the accommodating concept is appended at the front, e.g., X(Y,Z) used as shorthand for the concept of an X constituted of a Y and Z. The other reason for avoiding this format is the desire to align bracketing with concept realization. In the proposed approach, every bracketed entity realizes a concept.)

The shorthand is neither a prefix notation, nor a version of predicate logic then. More appropriate is to call it a programming language for concepts. The shorthand has a kinship with LISP, a functional programming language often used in AI (McCarthy et al., 1985/1962). LISP also uses prefix positioning to denote a particular deployment of a named entity, on which basis we might view the shorthand as a version of LISP in which function-calls evaluate to concepts. But, here again, the correspondence is not perfect. Deployment of a computational function is not the same thing as imposition of an encompassing concept. The treatment of arguments also differs. In a programming language such as LISP, arguments are given particular roles by positioning. This is not the case in the shorthand, where the encompassed elements form an unordered combination.

We can now return to the main objective, which is to demonstrate the expressive power of HCC by means of examples. An initial task is to show how varying the size of the accommodated combination can affect the outcome. Imagine we are provided with a base of four concepts: the concept of a flight, the concept of a drive, the concept of a journey and the concept of an excursion. The set of given concepts is then: flight, drive, journey, excursion. Since a journey can be made-up of a flight and a drive, a potential hierarchical combination is

[**journey** flight drive]

This places the concept of a journey into the accommodating role, with flight and drive as the accommodated combination. It realizes the idea of a journey made up of a flight and drive. Another possibility is

[**excursion** flight drive]

This expresses the subtly different concept of an excursion encompassing a flight and a drive. While the same combination is accommodated, the accommodating element differs, with the result that a different idea is obtained. What we obtain is the idea of a particular type of excursion, rather than a particular type of journey.

Also of interest are hierarchical combinations incorporating a single accommodated concept; e.g.,

[**journey** drive]

This constructs the idea of a journey consisting solely of a drive. Another combination yields the idea of a journey solely consisting of a flight:

[**journey** flight]

Minimal constructions like these are termed ‘singular accommodations’ or just ‘singles’ below. Intuitively, they can be seen as classifications. The second, for example,

can be seen to express the idea of a flight that is also classified as a journey. But notice there is no implication of either element being more general than the other. Singles do not define hyponyms. The signified relationship is hierarchical accommodation which, in the case of a single accommodated element, implies 'both' (i.e., mutual accommodation).

Singles are also inherently reversible. An X that is classified as a Y can also be seen as a Y classified as an X. Or, to put it another way, an X solely constituted of a Y can also be seen as a Y solely constituted of an X. By definition, therefore

[**X** Y] = [**Y** X]

Important to the expressive power of HCC is the possibility to place relational concepts in the encompassing role. With this done, the effect achieved is that of a relational schema. An illustrative example is

[**understanding** teacher lawyer]

This constructs the concept of an understanding encompassing a teacher and a lawyer (or an understanding between a teacher and a lawyer, as it would normally be described). The accommodating concept is implicitly an imposed relation, and what is constructed is a schema in result. But notice that no schema-making apparatus is involved. The concept of understanding provides the key contribution. Deployed in the accommodating role, it provides the 'glue' that holds the accommodated concepts together. The relational arrangement is captured purely by hierarchical combination—by taking one concept to encompass the other two.

Singles can be used to refine concepts of this type. For example, the following two-level structure might be specified:

[**understanding** [**agent** teacher] [**recipient** lawyer]]

This expresses the concept of an understanding encompassing a teacher and a lawyer, in which the teacher is classified[4] as agent, and the lawyer is classified as recipient. It builds the concept of a teacher understanding a lawyer, rather than the other way around. This begins to give a sense of how complex meanings of a compositional type can be realized by HCC.

The process of adding levels to a structure can continue as long as suitable concepts are available. We might, for example, write

[**development** [**understanding** [**agent** teacher] [**recipient** lawyer]]]

This adds a new level of meaning: The teacher's understanding of the lawyer is now classified as a development.

Use of singles, relational concepts and embedding increases the expressive power of hierarchical concept-combination, then. With all these possibilities put to use, the kinds of meaning we express using language are more easily obtained.[5] Consider the following, for example:

[**seeing.action** [**subject** John] [**object** [**definite.thing** book]]]

[4] Recall that singular accommodation implies mutual classification.

[5] Having recognized the way concepts can be compositionally constructed by HCC, it is natural to treat 'meaning', 'concept' and 'idea' as interchangeable terms.

This constructs the idea of a seeing.action encompassing John classified as subject, and a book classified as object, in which the book is also classified as a definite.thing. Taking into account the meaning of the subject and object concepts, what is composed is the idea of some individual John[6] seeing a definite book. It is the idea of an action done by a particular individual to a particular thing. We could express this in English by saying 'John sees the book'.

A more elaborate example is

[**yesterday.event**
  [**giving.action**
    [**subject** [**indefinite.thing** man]]
    [**object** bread]
    [**indirect.object** John]]]

Key to the meaning of this four-level construction is the first accommodated concept. Itself a structure, it expresses the idea of a giving action encompassing an indefinite man (classified as subject), bread (classified as object) and John classified as an indirect object. This yields the idea of an event in which an indefinite man gives bread to John. The event is then itself classified as a 'yesterday.event', i.e., an event occurring yesterday. The final product is the idea of a man giving bread to John yesterday. This is something we could express in English by saying 'yesterday a man gave bread to John'.

Some specialized forms of meaning, such as questions, can also be captured. Consider this, for example:

[**question**
  [[**event** drinking.action focal.thing]
    [**subject** [definite.thing teacher]]
    [**object** [**definite.thing** [**substance** water]]]]]

The central construction here is

[**event** drinking.action focal.thing]

This expresses the idea of an event encompassing a drinking.action and a focal.thing. Encompassed by this are water and a teacher, classified as subject and object respectively (and also as definite objects). This idea is then itself classified as a question. The final result is thus (the idea of) a question encompassing the idea of a definite teacher drinking some definite water. This is something we could express in English by asking 'Is the teacher drinking the water?'

These examples give a sense of the expressivity and productivity of hierarchical concept-combination. They reveal this to be language-like medium, in which complex meanings can be constructed. As noted, the generativity of the medium is unbounded in principle. If every construction that can be formed on some given base legitimizes at least one further construction, there are infinitely many meanings that can be derived. This is an important factor in the connection between HCC and language. Both are mechanisms which allow infinitely many meanings to be composed.

[6] For present purposes, names of individuals are taken to name the concept of the individual in question.

3. Merge

The focus can now return to Merge, and the question of how this operator is related to HCC. Informally, Chomsky characterizes the behavior of Merge as "Take two objects, make another object" (Chomsky in Boeckx, 2009: 52). More precisely, the operator "takes two syntactic objects α and β and forms the new object $\gamma = \{\alpha, \beta\}$" (Chomsky, 2001: 3). In building a hierarchical unit, Merge applies a particular constructive operation. The hierarchical unit constructed from α and β is defined to be $\{\alpha, \beta\}$. It is the *set* made up of the two constituents.

That HCC has the capacity to implement Merge can then be demonstrated. The hierarchical construction that Merge obtains by

$\gamma = \{\alpha, \beta\}$

can also be obtained by a hierarchical combination in which the concept of a set is the accommodating element, and α and β form the accommodated combination. The corresponding construction is

γ = [**set** α β]

In both cases, the hierarchical unit obtained specifies a set comprised of α and β.

HCC can also implement the operation known as 'internal Merge'. This involves re-merging the output of a construction with one of its constituents (e.g., in the case above, re-merging γ with α). What this produces is a two-level structure in which an element of the accommodated combination at the first level is also within the accommodated combination at the second. From the conceptual point of view, this is a perfectly legitimate state of affairs. Provided the constituents to which Merge applies can be viewed as concepts, HCC can implement the operation involved.

We can be confident that HCC has the capacity to implement Merge, then. Whether Merge has the capacity to implement HCC is less clear. One problem is the fact that HCC can by applied to combinations of any size, whereas the input to Merge is a binary set. This is straightforwardly overcome, however. Merge has the capacity to produce an input set of any size by hierarchically merging binary sets. Merge can be applied to sets of any size in this way.[7] The more serious obstacle is the fact that Merge appears to disallow variation in the accommodating concept. What is constructed in every case is a set.[8] Is there any way of deploying Merge that would allow variation in this?

The internal variant of the operation is potentially of use. As noted, internal Merge is the special case in which a merged entity is *re-merged* with one of its constituents. One constituent in particular is selected. In principle, this might be the means of differentiating an accommodating concept. In favour of this scheme is the way it connects the unbounded generativity of Merge to that of HCC. In the latter case, the unboundedness results from the compositional nature of the operation: Concepts are put together in a hierarchical assembly. What is obtained is inherently new (in the sense of being different to all the combined constituents), on which basis further constructions can be obtained in an ongoing way.

[7] My thanks to an anonymous reviewer for pointing this out.

[8] A hierarchical structure formed by application of Merge is a meronomy in this sense.

The unbounded generativity of Merge, on the other hand, is considered to stem from the way internal Merge can be re-applied any number of times. There is some debate as to whether this yields the right kind of generativity for language. Hinzen (2009: 137) makes the point that Merge cannot bring about 'categorial change', while Boeckx (2009) argues that its output is not genuinely new. In Boeckx's view, "once you combine two units, X and Y, the output is not some new element Z, but either X or Y." A related concern is the fact that a hierarchical assembly of object combinations can be reduced to (or expressed as) a non-hierarchical combination. This raises the question of whether the structure is hierarchical in the fullest sense. These concerns are not directly relevant, here, however. All that matters is that internal Merge is potentially the means of achieving the discrimination required for HCC. This also has the attraction of explaining the operation's unbounded generativity in a new way.

Subject to certain qualifications, then, it can be concluded that HCC can implement Merge, and Merge can implement HCC. The latter arrangement is problematic in one respect; but it is not unreasonable to assume the difficulty can be overcome. Taking the functional equivalence of the two mechanisms to be established, what should we then infer about the language system? Chomsky notes that either Merge or "some equivalent" is the minimal requirement for production of language (Chomsky, 2005: 11). Might HCC be the equivalent in question?

Assuming that it is induces a radically different conception of how language is produced. The hierarchical structure produced by HCC is a *semantic* object rather than a syntactic one. If this is the seed from which an utterance is derived, we have to assume there is some apparatus capable of turning it into a linguistic form. There has to be some mechanism which takes a semantic construct and derives from it a sequential-symbolic representation (i.e., an utterance). On this understanding, production of language would be accomplished on a 'semantics-first' basis, rather than a 'syntax-first' one.

Is there any way of putting this semantics-first arrangement on an operational footing? How could a hierarchical concept-combination with a certain meaning be translated into a semantically equivalent symbol sequence? The obvious way to accomplish this involves running semantic interpretation 'backwards'. For the language system to work there has to be knowledge of what symbols (i.e., words and morphemes) mean. This lexical knowledge is normally assumed to be invoked only at the stage of semantic interpretation. But the mapping can also be used to identify the symbol for a particular concept. The concept names used in a hierarchical concept-combination are potentially translated to symbols in this way.

But how are the the symbols to be got into a syntactically correct order? Here, the obvious solution is to make use of grammatical preferences. A sequential encoding of a concept structure, using symbols for the cited concepts, must distinguish the symbol that represents the accommodating concept. This has to be given a particular position in the sequence, otherwise the original construction cannot be inferred. Consider the typical case of a structure with two accommodated elements. This yields three symbols in all: one for the accommodating concept, and two for the accommodated concepts. Put into a sequence, these three symbols encode the original structure provided the accommodating symbol can be identified. This critical symbol has to be given a particular position in the sequence. It has to

go first, last or in the middle. Given there are two ways of ordering the accommodated symbols, this yields a total of $2 \times 3 = 6$ orderings in all. Letting X, Y and Z be the symbols in question, the six possible orderings are XYZ, XZY, YXZ, ZXY, YZX, ZYX.

The way grammatical preferences might be exploited then begins to become apparent. Languages typically have a preference to use a particular verb-phrase structure, where the six options are VSO, VOS, SVO, OVS, SOV, OSV. If the verb in a verbal construction is assumed to correspond to the accommodating element of a hierarchical concept-combination, the six verbal structures are then identical to the six symbol orderings for a typical concept structure. The ordering that the language is known to prefer can potentially be applied. Might this be enough to get symbols into the right order? Clearly, by itself, it would not. It fails to deal with cases involving anything other than two accommodated concepts. But the general idea of applying grammatical preferences can be put to use in different ways. This approach can be the means of obtaining syntactically correct outputs in a number of cases, as the examples of the next section illustrate.

4. Symbolic Encoding by Application of Grammatical Preferences

To illustrate the way grammatical preferences can be used to translate a semantic form (a hierarchical concept-combination) into a linguistic form (a symbolic sequential encoding) with the same meaning, it is convenient to return to one of the HCC examples used above:

[**seeing.action** [**subject** John] [**object** [**definite.thing** book]]]

Recall that this constructs the idea of John seeing a particular book, i.e., an idea that can be expressed in English by 'John sees the book'. How can the HCC be translated into this expression? Grammatical preferences of English that apply in this context include (1) a preference for SVO ordering, (2) a preference to identify subject and object by ordering, and (3) a preference for head-initial organization in simple phrases. These can be captured by means of the following three rules.

2 1 3 $\overset{a}{\longleftarrow}$ [= subject object]
2 $\overset{b}{\longleftarrow}$ [subject/object =]
1 2 $\overset{c}{\longleftarrow}$ [= =]

These rules are notated on a right-to-left basis, with labels placed over the arrows for easy reference. Each rule shows the preferred ordering for a particular semantic construction (i.e., HCC). The ordering appears to the left of the arrow, and the semantic construction to the right. The first rule (labeled $\overset{a}{\longleftarrow}$) encodes the preference for SVO organization; the second (labeled $\overset{b}{\longleftarrow}$) encodes the preference to encode subject and object classifications by means of ordering, while the third (labeled $\overset{c}{\longleftarrow}$) encodes the preference for head-initial organization in simple phrases.

The notation works in the following way. Rule $\overset{a}{\longleftarrow}$ applies to any hierarchical concept-combination which conforms to the shorthand:

[= subject object]

The structure can have anything as its accommodating element (the '=' is a wild-

card) but the accommodated combination must *comprise* subject and object concepts. (Recall that, in the shorthand, accommodated concepts have no order.) A subject concept is specified either by name (i.e., as 'subject') or as a construct for which 'subject' is encompassing either explicitly or implicitly. This means [**subject** John] is a subject concept, as is [**definite.thing** [**subject** John]].

The numbers on the left of a rule specify the way symbols should be ordered. Each number indexes an element of the specification on the right, while its *position* says where symbols arising for that element should be placed. Rule $\overset{a}{\longleftarrow}$ has '2 1 3' on the left. This means symbols arising for whatever matches the 2nd element should be placed first, followed by symbols for whatever matches the 1st element, followed by symbols for whatever matches the 3rd element. Given the structures it can match to (and the assumption that verbs signify accommodating concepts) this rule captures the preference for SVO ordering.

Rule $\overset{b}{\longleftarrow}$ uses '/' to denote alternatives. The specification matches any structure in which the first element is either a subject or object concept. The designation on the left is just '2', meaning only the encompassed element is symbolized. Given the structures it can match to, this rule captures the preference for expressing subject and object classifications implicitly. The final rule deals with any single (i.e., any concept with a single accommodated element). It specifies that the symbol(s) for the accommodating element should be placed before the symbol(s) for the accommodated element. Given its coverage, this rule expresses the preference for head-initial organization in simple phrases. Earlier rules take precedence, so $\overset{a}{\longleftarrow}$ has priority over $\overset{b}{\longleftarrow}$, which has priority over $\overset{c}{\longleftarrow}$.

With these rules defined, a syntactically valid expression of the HCC's meaning can then be derived, provided the concepts cited in the structure are symbolized as follows.

book ← book
the ← definite.thing
John ← John
sees ← seeing.action

Each line here specifies the symbol to be used for a particular concept. The concept is specified to the right of the '←', and the symbol (word or morpheme) to the left. The symbol specified for the concept of a book is defined to be *book*, for example.

Translation of the HCC, in accordance with the specified rules, then proceeds as follows. At the start of the process, the object to be translated is the conceptual structure itself. The only rule whose right-hand-side matches this object is $\overset{a}{\longleftarrow}$, which is an organizational rule. Applying it has the effect of decomposing the object into three constituents: the referents of the specified ordering. These objects are then translated in the same way. Whenever an object to be translated matches the right-hand-side of a lexical rule, it is immediately translated to the corresponding symbol. Once all the constituents realized by applying an organizational rule to an object have been translated, their encodings are put in the specified order, and this sequence becomes the translation of the object itself. Eventually, the entire conceptual structure is rendered into a sequential-symbolic form.

The processing can be shown schematically as follows.

→ [**seeing.action** [**subject** John] [**object** [**definite.thing** book]]]
→ [**subject** John]
← *John* (John)
$\overset{b}{\longleftarrow}$ *John*
← *sees* (seeing.action)
→ [**object** [**definite.thing** book]]
→ [**definite.thing** book]
← *the* (definite.thing)
← *book* (book)
$\overset{c}{\longleftarrow}$ *the book*
$\overset{b}{\longleftarrow}$ *the book*
$\overset{a}{\longleftarrow}$ *John sees the book*

This listing uses indentation to visualize recursion. Each line represents application of a rule. For each application of a lexical rule, there is a line which ends with the relevant concept name. For example, translation of the concept name 'book' to the symbol *book* is denoted by the line

← *book* (book)

For each application of an organizational rule, there is a line showing the concept that is processed and—at with same indentation below—a second line showing the symbol sequence assembled. Use of rule $\overset{b}{\longleftarrow}$ to turn [**definite.thing** book] into *the book* thus has an upper line of the form

→ [**definite.thing** book]

and a lower line of the form

$\overset{b}{\longleftarrow}$ *the book*

At completion of processing, the final translation obtained is *John sees the book*, which is a syntactically valid way of expressing the meaning of the original conceptual structure.

This example illustrates how a hierarchical conceptual structure can, by application of grammatical preferences, be translated to a syntactically valid utterance with the same meaning. It gives a sense of how HCC might fulfil the function of the equivalent of Merge that Chomsky envisages. But notice the very different conception of language production that arises. The process is not seen to be shaped by knowledge of syntax. It is seen to stem from a capacity for hierarchical concept-combination, coupled with an ability to apply grammatical preferences in the derivation of symbolic encodings. Production of language is seen to be a kind of assembly-line process, the first stage of which is construction of a semantic object in the conceptual system, and the second stage of which is encoding of that object into a sequential symbolic form.

More complex examples can also be assembled. All the HCC examples of Section 2 can be put to use in this way. We can also vary the language in which the output comes to be expressed. Consider again the utterance 'John read the letter'. Translated into Japanese, this becomes

Johnga tegamio yonda.

For an assembly-line analysis of this sentence we require a hierarchical concept-combination which expresses the correct meaning. As previously noted, a construction with the meaning of 'John read the book' is

[[**past.behavior** reading.action] [**subject** John] [**definite.thing** letter]]

How can this be translated into syntactically valid Japanese? Grammatical preferences of Japanese in this context include (1) a preference for SOV ordering, and (2) a preference for head-final organization in simple phrases. These can be captured as follows:

2 3 1 $\xleftarrow{a}$ [= subject definite.thing]
2 1 $\xleftarrow{b}$ [= =]

Given the concepts it can match to, use of '2 3 1' in rule $\xleftarrow{a}$ expresses the preference for SOV organization, while rule $\xleftarrow{b}$ expresses the preference for head-final organization in simple phrases. A syntactically valid expression is then obtained, provided the cited concepts are symbolized as follows:[9]

John ← John
tegami ← letter
yon ← reading.action
da ← past.behavior
ga ← subject
o ← definite.thing

Translation of the conceptual structure then proceeds as follows:

→ [[**past.behavior** reading.action] [**subject** John] [**definite.thing** letter]]
  → [**subject** John]
    ← *John* (John)
    ← *ga* (subject)
  $\xleftarrow{b}$ *John ga*
  → [**definite.thing** letter]
    ← *tegami* (letter)
    ← *o* (definite.thing)
  $\xleftarrow{b}$ *tegami o*
  → [**past.behavior** reading.action]
    ← *yon* (reading.action)
    ← *da* (past.behavior)
  $\xleftarrow{b}$ *yon da*
$\xleftarrow{a}$ *John ga tegami o yon da*

[9] Lexical preferences are derived from the analysis of (Kuno, 1973: 10).

(i) *John-ga* *tegami-o* *yon-da.*
John-SUBJ letter-OBJ read-PST
'John read the letter.'

See also WALS, Ch. 82, Ex. 2.

With word-breaks imposed, the output is *Johnga tegamio yonda*, which is the desired Japanese sentence.

Another example taken from Section 2 is

[**yesterday.event**
  [**giving.action**
    [**subject** [**indefinite.thing** man]]
    [**object** bread]
    [**indirect.object** John]]]

Recall that this builds the idea of an indefinite man giving bread to an individual, John, at a particular point in time, namely yesterday. The meaning is one we could express in English by saying 'yesterday a man gave bread to John'. A sentence from the Suriname language of Arawak with a not dissimilar meaning is

Miaka aba wadili sika khali damyn.

This means 'yesterday a man gave cassava bread to me'. To capture this meaning, the conceptual structure above needs to be modified in two ways. The recipient of the action needs to be specified as 'me' rather than 'John', and the object needs to be 'cassava.bread' rather than 'bread'. This produces

[**yesterday.event**
  [**giving.action**
    [**subject** [**indefinite.thing** man]]
    [**object** cassava.bread]
    [**indirect.object** me]]]

How can this structure be translated into Arawak? Grammatical preferences of this language in this context include (1) a preference for SVO organization; (2) a preference for dealing with subject and object by ordering; (3) a preference for head-final organization in simple phrases denoting an indirect object; and (4) a preference for head-final organization elsewhere. These can be captured by the following four rules.

2 1 3 4 $\leftarrow^{a}$ [= subject object indirect.object]
2 $\leftarrow^{b}$ [subject/object =]
2 1 $\leftarrow^{c}$ [indirect.object =]
1 2 $\leftarrow^{d}$ [= =]

With these rules defined, a syntactically valid Arawak expression is obtained provided cited concepts are symbolized as follows:[10]

[10] All preferences derived from the analysis of (Pet, 1987).

(i)

Miaka	*aba*	*wadili*	*sika*	*khali*	*da-myn.*
yesterday	INDEF	man	give	cassava.bread	1SG-to

'Yesterday a man gave cassava.bread to me.'

See also WALS, Ch. 84, Ex. 4.

khali ← cassava.bread
sika ← giving.action
aba ← indefinite.thing
myn ← indirect.object
wadili ← man
da ← me
miaka ← yesterday.event

Mapping of the conceptual structure then proceeds as follows. (Notice some lines are truncated.)

→ [**yesterday.event** [**giving.action** [**subject** [**indefinite.thing** man]] ...
  ← *miaka* (yesterday.event)
  → [**giving.action** [**subject** [**indefinite.thing** man]] [**object** ...
    → [**subject** [**indefinite.thing** man]]
      → [**indefinite.thing** man]
        ← *aba* (indefinite.thing)
        ← *wadili* (man)
      $\overset{d}{\leftarrow}$ *aba wadili*
    $\overset{b}{\leftarrow}$ *aba wadili*
    ← *sika* (giving.action)
    → [**object** cassava.bread]
      ← *khali* (cassava.bread)
    $\overset{b}{\leftarrow}$ *khali*
    → [**indirect.object** me]
      ← *da* (me)
      ← *myn* (indirect.object)
    $\overset{c}{\leftarrow}$ *da myn*
  $\overset{a}{\leftarrow}$ *aba wadili sika khali da myn*
$\overset{d}{\leftarrow}$ *miaka aba wadili sika khali da myn*

With word-breaks imposed, the output is *Miaka aba wadili sika khali damyn*, which is the desired Arawak sentence.

This example can also be used to give a sense of how the approach might explain movement—the "curious but ubiquitous phenomenon of displacement in natural language" as Chomsky describes it (Chomsky, 2009a: 31). In the statement 'yesterday a man gave cassava bread to me', the initial word can be moved to final position without affecting the meaning. The statement then becomes 'a man gave cassava bread to me yesterday'. If we assume that in an English version of the above, two organizational rules are invoked by the structure [**yesterday.event** ...], one having '1 2' as its ordering, and the other having '2 1', the selection between them is potentially made at random. On this basis, the optionality of the two forms can be explained by saying either way of formulating the sentence is potentially derived.

To complete this series of examples, we should also look at

[**question**
[[**drinking.action** focal.thing]
[**subject** [definite.thing teacher]]
[**object** [**definite.thing** [**substance** water]]]]]

Recall that this builds an idea that would be expressed in English by asking the question 'Is the teacher drinking the water?' Say we would like to develop an analysis of this question expressed in German. The question then takes the form

Trinkt der lehrer das wasser?

How can this German question be derived from the conceptual structure? Grammatical preferences of German in this context include (1) a preference to encode subject and object classifications by ordering alone; (2) the preference for VSO organization given a meaning classified as a question; (3) the preference for SVO organization otherwise, and (4) the preference for head-initial organization in simple phrases. These can be encoded by the following four rules.

2 $\xleftarrow{a}$ [subject/object =]
2 3 4 $\xleftarrow{b}$ [question [= subject object]]
2 1 3 $\xleftarrow{c}$ [= subject object]
1 2 $\xleftarrow{e}$ [= =]

With these definitions given, a syntactically valid, German expression of the construct's meaning is obtained provided cited concepts are symbolized as follows:[11]

lehrer ← teacher
wasser ← water
trink ← drinking.action
das ← [**definite.thing** substance]
der ← definite.thing
t ← focal.thing

Notice the use of a structured specification in the case of *das*. This is required to capture the restricted range of this determiner.

Mapping of the conceptual structure then proceeds as follows:

→ [**question** [[**drinking.action** focal.thing] [**subject** ...
→ [**drinking.action** focal.thing]
← *trink* (drinking.action)
← *t* (focal.thing)
$\xleftarrow{e}$ *trink t*
→ [**subject** [**definite.thing** teacher]]
→ [**definite.thing** teacher]

[11] Lexical preferences are derived from the analysis of (Dryer and Haspelmath, 2011, Ch. 116, Ex. 6).

(i) *Trink-t* *der* *Lehrer* *das* *Wasser?*
drink-3SG DEF teacher DEF water
'Is the teacher drinking the water?'

← *der* (definite.thing)
← *lehrer* (teacher)
$\leftarrow^{e}$ *der lehrer*
$\leftarrow^{a}$ *der lehrer*
→ [**object** [**definite.thing** [**substance** water]]]
← *das* ([definite.thing substance])
← *wasser* (water)
$\leftarrow^{a}$ *das wasser*
$\leftarrow^{b}$ *trink t der lehrer das wasser*

Once standard word-breaks and capitalization have been imposed, the output is found to be the desired German question: *Trinkt der Lehrer das Wasser?*

If we remove the question classification in the original conceptual form, only the inner structure remains:

[[**drinking.action** focal.thing]
[**subject** [definite.thing teacher]]
[**object** [**definite.thing** [**substance** water]]]]

This realizes a meaning we might express in English by the statement 'the teacher is drinking the water.' Applying the rules to this reduced structure then invokes the default SVO ordering, with the effect of placing the verb between rather than before subject and object. The processing is as follows:

→ [[**drinking.action** focal.thing] [**subject** ...
→ [**subject** [**definite.thing** teacher]]
→ [**definite.thing** teacher]
← *der* (definite.thing)
← *lehrer* (teacher)
$\leftarrow^{e}$ *der lehrer*
$\leftarrow^{a}$ *der lehrer*
→ [**drinking.action** focal.thing]
← *trink* (drinking.action)
← *t* (focal.thing)
$\leftarrow^{e}$ *trink t*
→ [**object** [**definite.thing** [**substance** water]]]
← *das* ([definite.thing substance])
← *wasser* (water)
$\leftarrow^{a}$ *das wasser*
$\leftarrow^{c}$ *der lehrer trink t das wasser*

With word-breaks and capitalization imposed, the output is then *der Lehrer trinkt das Wasser*, which expresses the assertion 'the teacher is drinking the water'.

The examples of this section demonstrate that, at least in certain cases, syntactically valid outputs can be obtained by mapping conceptual structures to symbol sequences in accordance with grammatical preferences. Is there any reason to believe this might be possible more generally? Can the assembly-line model be

applied in more than just this handful of cases? There is a case for thinking the approach may be no less general than conventional syntactic analysis. The argument is based on the idea that any syntactical analysis can be converted into an assembly-line analysis. This is done by decomposing the syntactic analysis into two parts, one dealing with hierarchical structure, and the other dealing with sequential structure, where the former is an HCC, and the latter are the rules defining grammatical preferences and symbolization. The decomposition separates the specifically hierarchical aspect of the syntactic analysis from the specifically sequential specification.

The decompositional process can be illustrated using 'John sees the book' again. This utterance takes the form of a verb phrase in which 'sees' is the verb, 'John' is the subject, and 'the book' is the object. The structure can be analyzed as follows:

$$(_{VP}\ \text{sees}(_{N}\ \text{John})\ (_{NP}\ (_{DET}\ \text{the})(_{N}\ \text{book})))$$

To decompose this analysis into two parts, we proceed as follows. First, we replace each grammatical constituent with a semantically equivalent conceptual constituent. In the simplest case, the constituent is a content word, and the replacement is just the corresponding concept name. The two content words here are 'John' and 'book', but both can be seen as concept names in their own right, so no change is needed. The second step involves replacing each grammatical construction with a semantically equivalent conceptual construction. This is a hierarchical concept-combination in which the encompassing concept has the meaning that the grammatical construction imposes on its subordinate elements. Or, to put it another way, the encompassing concept is whatever idea is realized by the construction, once the constituents are abstracted away.

In this case there are two such constructions: the verb phrase and the noun phrase. In the latter, the word 'the' is an article (an ART.DEF). As this is the definite article, a particular meaning is implied—that what is referred to is a definite as opposed to indefinite book. The effect is to impose the meaning of the book being a definite thing. The construct is thus replaced with an HCC which imposes the relevant classification:

[**definite.thing** book]

This has the effect of classifying[12] the book as a definite.thing.

The verb 'sees' imposes the meaning (in the same sense as above) of a seeing action on the subordinate subject and object. This is thus replaced with the construction

[**seeing.action** John [**definite.thing** book]]

This expresses the idea of a seeing action encompassing John and a definite book. Still needed is discrimination of subject and object. This is where the grammatical structure captures meaning by the ordering of branches. The SVO ordering of the verb phrase has the effect of classifying 'John' as subject, and 'the book' as object. Branches in the conceptual structure are unordered, so for the conversion, we need to add the relevant classifications. The structure then becomes

[12] Recall that mutual accommodation implies mutual classification.

[**seeing.action** [**subject** John] [**object** [**definite.thing** book]]]

This is the structure that we already know expresses the meaning of 'John sees the book'. The semantic part of the assembly-line analysis can be derived from the syntactic analysis in this way. The other part of the analysis then comprises the grammatical preferences, both lexical and organizational, that produce the sequential symbolization.

The semantics-first approach, which results from assuming HCC fulfils the function of what Chomsky calls 'some equivalent' of Merge, may have a reasonable degree of generality, then. If a conventional syntactic analysis can be translated into a semantics-first account, the two approaches have the same explanatory range. What seems most problematic about the semantics-first model is that it eliminates the role normally attributed to syntactic knowledge. But the idea that languages *have* syntactic structure is retained. The assumption is that this can be seen as the confluence of two shaping influences, one being HCC, and the other being the grammatical preferences of the language in question. On this basis, the assembly-line model is closer to the norm than it may seem. Fully acknowledged is the fact that languages have structure, and that taxonomizing this structure reveals syntax. What is added is the observation that this structure may be the result of an interaction between two forces.

5. Discussion

Taking all the evidence into account, what can be concluded? Some aspects of the situation are certainly beyond dispute. At the heart of the language system there has to be a mechanism of hierarchical construction which assembles expressions of arbitrary richness and complexity. As Chomsky notes, "Either Merge or some equivalent is a minimal requirement." (Chomsky, 2005: 11-12) The capacity to combine concepts hierarchically is also central to the conceptual system. A mechanism of hierarchical concept-combination has to exist.[13] Both Merge and HCC must, then, be implemented in the mind.

What is at issue is their relationship. Hauser et al. characterize the functionality of Merge as "hierarchical, generative, recursive, and virtually limitless with respect to its scope of expression" (cf. Hauser et al., 2002: 1569). Precisely the same description can be given to HCC. An equivalence between the two mechanisms is suggested and, on close inspection, this is found to be nearly perfect. HCC can reproduce Merge, and subject to certain assumptions, Merge can do the same for HCC. In consequence, HCC might be the equivalent of Merge that Chomsky envisages. Assuming this to be so results in the semantics-first interpretation of language production, as described above.

If we set the semantics-first model aside, two possibilities then remain. One is that Merge and HCC are completely disconnected. This is a viable option. Merge can be seen to play its usual role in the language system, while HCC operates separately in the conceptual system. This conforms to the idea of the two systems

[13] It is not claimed here that this is the only way conceptual structure can be built. Reinhart's observation that ordering of constituents generally plays no role in conceptual structure (Reinhart, 1976) is consistent with HCC being the constructive mechanism implied, however.

being functionally independent. But it does entail assuming significant duplication of cognitive resources. We have to assume the mind possesses two functionally equivalent ways of constructing meanings compositionally. There must be two distinct generative mechanisms, and they must be resourced independently.

The other possibility is that one of the two mechanisms provides constructive services to the other on a client-server basis. The advantage of this is that it avoids assuming duplication of resources, while retaining the standard conception of language being produced on a modular 'syntax-first' basis. Challenging questions do arise, however. For this arrangement to work, the interface between the language and conceptual systems must have the capacity to mediate the provision of constructive services. How is this accomplished? Whichever operator is the 'server', we have to assume it can be manipulated in a way that allows both constructions of syntax, and constructions of HCC to be obtained. How this dual functionality is operationalized becomes a critical issue.

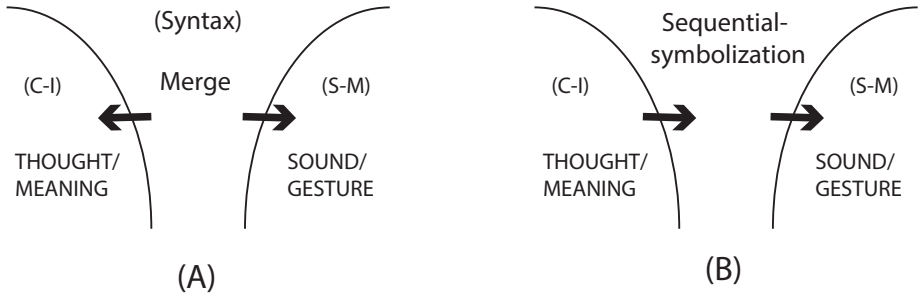

Figure 1: Alternative conceptions of the Merge-HCC relation: (A) syntax-first processing in the standard Minimalist architecture; (B) semantics-first processing.

If we rule out the idea of Merge and HCC being completely disconnected on the grounds that this entails too great a duplication of mental resources, what remains is either the syntax-first arrangement, or the previously described semantics-first arrangement. These are illustrated schematically in Figure 1. Panel (A) represents the syntax-first case. This is essentially the standard Minimalist architecture, with Merge as the central element. There is one interface which conveys constructions to the conceptual system (C-I), and another which externalizes them via sound and gesture (S-M). The only modification is the assumption that Merge provides constructive services to HCC, or vice versa, and that the Merge/C-I interface mediates this.

Panel (B) illustrates the semantics-first alternative. Here, the hierarchical structure underlying an utterance is seen to be constructed in the conceptual system, by hierarchical concept-combination. The flow of information is left-to-right. The conceptual system (C-I) is assumed to send hierarchical structures to an intervening symbolization module, which then uses grammatical preferences to derive syntactically well-formed outputs. These are then sent for externalization via S-M in the usual way.

Like the syntax-first model, the semantics-first model has a mix of pros and cons, some of which have already been discussed. The key advantage is that it avoids the need for linguistic Merge. HCC is taken to be the Merge-equivalent that Chomsky envisages. This makes compositional thought the generative system behind language, a strategy which Hinzen notes would be appealingly parsimonious (Hinzen, 2012: 637). Another positive feature of the interpretation, not previously mentioned, is its ability to reconcile views on language and thought. It has been seen that HCC is a medium in which compositional meanings can be constructed, and that, like Merge, it is limitless in its scope of expression. Some theorists have doubted whether there is any medium other than language which is compositional in this way (e.g. Chomsky, 2007b; Hinzen, 2012). The existence of HCC shows there is, and this has implications for the debate about how language and thought are related.

The debate has particularly focused on the issue of whether thought or language is prior. With the structural forms of language assumed to be defined by grammar (syntax), the key question has been whether it is thought which shapes grammar, or grammar which shapes thought. Theorists inclined towards the former arrangement include (Bates and MacWhinney, 1979, 1987; Bickerton, 1990; Fodor, 2001; Elman, 2004; Kirby, 1999; Jackendoff, 2002; Tomasello, 2003; Clair et al., 2009; Chater and Christiansen, 2010; Tomasello, 2008). Theorists inclined to the view that it is grammar which shapes thought include (Fitch and Chomsky, 2005; Chomsky, 2012, 2009b; Hauser, 2009; Hinzen, 2009).

The existence of HCC raises the possibility of an intermediate position. It allows that thought might shape grammar at the same time as grammar shapes thought. Under the semantics-first arrangement, the hierarchical forms of HCC are seen as shaping the hierarchical structure of utterances, while their sequential-symbolic form is seen to be shaped in another way—by grammatical preferences. On this view, thought shapes the *hierarchical* forms of language only. The idea of thought shaping language in this partial sense does have some support in the literature. Jackendoff, for example, observes that meaning would "be the first generative component of language to emerge" (Jackendoff, 2003: 664), due to being a "combinatorial system independent of, and far richer than, syntactic structure" (Jackendoff, 2002: 123). It also conforms to Hauser's view of language as a "mind-internal computational system designed for thought and often externalized in communication" (Hauser, 2009: 74).

But the claim that grammar shapes thought is then also validated to some degree. Grammar is seen to be a formulation in which hierarchical constructions of thought obtain a sequential form. On this basis, the position that "thought itself is fundamentally linguistic" (Corballis, 2014: xiii) is not invalid. Nor is Hinzen's observation that in use of language "we see new forms of meaning arising in ways that exactly reflect a narrow range of grammatical operations and options" (Hinzen, 2012: 646). On the semantics-first model, the structures which convey these meanings are seen to be modulated by the ways in which ordering is imposed. But this still allows that, in some sense, "the generative system of language underlies and is actually indistinguishable from the generative system that powers abstract thought" (Hinzen, 2009: 125). Grammar is seen to express the compositional structures of thought, accommodating Chomsky's observation that "language evolved,

and is designed primarily as an instrument of thought" (Chomsky, 2009a: 29).

In other respects, the semantics-first model is less attractive. There is no implementational question, since HCC can fully perform the function of Merge. But the model changes the conception of how language is produced in a radical and potentially unacceptable way. The indications are that syntactically well-formed utterances can be produced by mapping hierarchical concept-combinations to symbol sequences in accordance with context-specific grammatical preferences. This explains how an inherently unordered object—an HCC—can give rise to an inherently ordered one—a sequence of symbols. But it also has the effect of eliminating conventional syntax from the account. Explicit syntactic rules are seen to play no role.

This *might* be seen as advantageous. The requirement to explain how the language system adheres to, and applies rules of syntax is dissolved, as is the requirement to discover what the rules are. Instead, there is a requirement to explain how grammatical preferences can turn conceptual structures into symbol sequences. Assuming, fewer rules are required for this, the effect is to simplify the account. This is one way the model might be seen as more parsimonious. There are others. Semantic and syntactic processing are normally seen as separate. The language system is assumed to perform syntactic processing independently. Hinzen calls this "the spirit of the autonomy of syntax" (Hinzen, 2012: 638). Consequently, there is a need to explain how semantic processing comes to influence language production, an objective that has been called the "final frontier" of linguistics (Fitch, 2009: 306). Under the semantics-first model, this requirement is eliminated. Semantic processing is now the first stage in an assembly-line process from which syntactically correct utterances emerge. This makes semantics an integral part of syntax. At the same time, it makes the grammatical preferences which shape the second stage functionally critical. This might also be seen as beneficial, however, as it answers a puzzling question. Grammatical preferences (e.g. for head-initial organization) have often been seen as problematic from the explanatory point of view (Polinsky, 2012).

The semantics-first model is not without explanatory benefits, then. The problem is that most of them can also be seen as costs. The model can be seen as failing to respect the autonomy of syntax, and as ignoring the distinction between syntax and semantics. By making semantics an integral part of syntax, it effectively eliminates the traditional idea of a syntax/semantics interface. The model can also be seen as blurring the distinction between syntax and morphology, and as negating the principle that grammatical preferences are functionally neutral. It can only be adopted, it seems, at the price of flying in the face of established theory.

There is no clear winner between the syntax-first and semantics-first interpretations, then. Both present a mix of advantages and disadvantages. Can we escape the impasse by assuming the existence of a more primitive operator, from which both Merge and HCC are descended?[14] Might it be that Merge and HCC have a common ancestor, without being directly related? It is difficult to see how there could be an ancestor of HCC, even in principle. Any ancestor from which HCC is derived must subserve the capacity to combine concepts in a way that makes one the accommodating element with respect to the others. But in that case, the an-

[14] My thanks to two anonymous reviewers for suggesting this possibility.

cestor is HCC itself. There seems to be no way to reduce this operation to a more primitive form. Merge can perhaps, be seen as a descendent of HCC, since the operation it performs is a special case of HCC. But the existence of a common ancestor seems to be ruled out.

6. Concluding Comments

The Minimalist approach sees Merge as the generative mechanism behind language. But it also acknowledges that any unbounded system of hierarchical construction might suffice. Hierarchical concept-combination, it seems, fits the bill so could be all that is needed. But this turns the language system on its head. The generative mechanism behind language is now in the *conceptual* system and, more awkwardly still, is a generator of semantic rather than syntactic structure. It is a medium for assembling arbitrarily complex meanings by compositional thought. The only way to accommodate this is to change the central role of the language system altogether. No longer can this be the building and interpreting of syntactic structure. Instead, it has to be sequential-symbolic encoding/decoding of hierarchical structure native to the conceptual system.

This, the semantics-first or assembly-line model, is one way in which Merge and HCC might be related. Arguably, the most interesting option, it is also the one which produces most epistemic upheaval. The other two possibilities conform more to established theory. The first is that Merge and HCC are completely disconnected. The second is that one of these mechanisms uses the other on a client-server basis. The latter raises thorny questions about implementation, while the former entails assuming an inherently implausible duplication of resources. On the present evidence, then, it is hard to establish which of the three arrangements is closest to the truth.

Empirical investigation could certainly pay dividends in this situation. The semantics-first model gives rise to specific predictions. Syntactically correct utterances are seen to stem from the way HCC interacts with sequential symbolization. Knowledge of syntax plays no role. But if knowledge of sequencing conventions, combined with a capacity for HCC, suffices to produce utterances conforming to syntactic rules, linguistic performance should run well ahead of syntactic evidence. Language learners should be seen to produce utterances that conform to syntactic rules without ever encountering any examples of the rules in question. Experimentation probing this capacity is one way of empirically testing the semantics-first model, then. (According to some theorists, evidence of this capacity already exists in the form of poverty-of-stimulus effects. The claim remains controversial, but should it ultimately be validated, this would favour the semantics-first model.)

Another prediction of the semantics-first model stems from its commitment to the idea of assembly-line processing. If language production involves two distinct stages of processing, the first semantic, the second non-semantic, this should be reflected in the neurological evidence. Under the standard model of language production, syntax is seen as relatively autonomous. This entails that semantic processing should follow or co-occur with non-semantic processing. Under the semantics-first model, it is the other way around. Semantic processing must be complete before non-semantic processing can even begin. The processing in the

brain that mediates production of an utterance should exhibit two stages, then: the first inherently semantic, the second inherently non-semantic. If evidence of this division already exists, or can be obtained, that would favour the semantics-first interpretation. Resolution of the question of how Merge and HCC are related in the mind is likely to come from empirical investigation, then. Assessment of evidence that already exists may also be able to shed some light. It is hoped that future work will make progress in these ways.

References

Bates, Elizabeth & Brian MacWhinney. 1979. A functionalist approach to the acquisition of grammar. In Elinor Ochs & Bambi Schieffelin (eds.), *Developmental Pragmatics*, 167–209. Academic Press.

Bates, Elizabeth & Brian MacWhinney. 1987. Competition, variation, and language learning. In Brian MacWhinney (ed.), *Mechanisms of Language Acquisition*, 157–93. Erlbaum.

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

Boeckx, Cedric. 2009. The nature of merge: Consequences for language, mind, and biology. In Massimo Piattelli-Palmarini, Juan Uriagereka & Pello Salaburu (eds.), *Of Minds and Language: A dialogue with Noam Chomsky in the Basque Country*, 44–57. Oxford University Press.

Carey, Susan. 2009. *The Origin of Concepts*, Oxford University Press.

Chater, Nick & Morten H. Christiansen. 2010. Language acquisition meets language evolution. *Cognitive Science 34*(7). 1131–1157.

Chomsky, Noam. 2001. Derivation by phase. *Ken Hale: A Life in Language*, The MIT Press. 1–52.

Chomsky, Noam. 2005. Three factors in language design. *Linguistic Inquiry*, *36*. 1–22.

Chomsky, Noam. 2007a. Biolinguistic explorations: Design, development, evolution. *International Journal of Philosophical Studies 15*. 1–16.

Chomsky, Noam. 2007b. Of minds and language. *Biolinguistics 1*. 9–27.

Chomsky, Noam. 2009a. Opening remarks. In Massimo Piattelli-Palmarini, Juan Uriagereka & Pello Salaburu (eds.), *Of Minds and Language: A dialogue with Noam Chomsky in the Basque Country* , 13–43. Oxford University Press.

Chomsky, Noam. 2009b. Concluding remarks. In Massimo Piattelli-Palmarini, Juan Uriagereka & Pello Salaburu (eds.), *Of Minds and Language: A dialogue with Noam Chomsky in the Basque Country*, 379–409. Oxford University Press.

Chomsky, Noam. 2012. *The Science of Language: Interviews with James McGilvray*, Cambridge: Cambridge University Press.

Corballis, Michael C. 2014. *The Recursive Mind: The Origins of Human Language, Thought, and Civilization*, Princeton and Oxford: Princeton University Press.

Costello, Fintan J. & Mark T. Keane. 2001. Testing two theories of conceptual combination: Alignment versus diagnosticity in the comprehension and production of combined concepts. *Journal of Experimental Psychology: Learning, Memory and Cognition 27*(1). 255–271.

Dryer, Matthew S. & Martin Haspelmath. 2011. *The World Atlas of Language Structures Online*, Munich: Max Planck Digital Library. Package online at http://wals.info/ Accessed on 2013-04-06.

Elman, Jeffrey L. 2004. An alternative view of the mental lexicon. *Trends in Cognitive Sciences 8*. 201–206.

Fitch, Tecumseh W. & Chomsky, Noam. 2005. The evolution of the language faculty: Clarifications and implications. *Cognition 97*(2). 179–210.

Fitch, Tecumseh W. 2009. Prolegomena to a future science of biolinguistics. *Biolinguistics 3*(4). 283-320.

Fodor, Jerry A. 2001. Language, thought, and compositionality. *Mind & Language 16*(1). 1–15.

Gentner, Dedre. 1983. Structure-Mapping: A theoretical framework for analogy. *Cognitive Science 7*. 155-170.

Hampton, James A. 1991. The combination of prototype concepts. In Paula J. Schwanenflugel (Ed.), *The Psychology of Word Meanings*, Hillsdale, N.J: Lawrence Erlbaum Associates.

Hampton, James A. 1997. Conceptual combination: Conjunction and negation of natural concepts. *Memory and Cognition 25*. 888–909.

Hampton, J. 2011. Conceptual combination and fuzzy logic. In Radim Bělohlavek & George J. Klir (eds.), *Concepts and Fuzzy Logic*, 209–232. MIT Press.

Hauser, Marc D., Noam Chomsky & Tecumseh W. Fitch. 2002. The faculty of language: what is it, who has it, and how did it evolve? *Science 198*. 1569-1579.

Hauser, Marc D. 2009. Evolingo: The nature of the language faculty. In Massimo Piattelli-Palmarini, Juan Uriagereka & Pello Salaburu (eds.), *Of Minds and Language: A dialogue with Noam Chomsky in the Basque Country*. 74–84, Oxford University Press.

Hinzen, Wolfram. 2009. Hierarchy, Merge, and truth. In Massimo Piattelli-Palmarini, Juan Uriagereka & Pello Salaburu (eds.), *Of Minds and Language: A dialogue with Noam Chomsky in the Basque Country*, 124–141. Oxford University Press.

Hinzen, Wolfram. 2012. The philosophical significance of Universal Grammar. *Language Sciences 34*. 635-649.

Jackendoff, Ray. 2002. *Foundations of Language: Brain, Meaning, Grammar, Evolution*, Oxford University Press.

Jackendoff, Ray. 2003. Précis to Foundations of Language: Brain, Meaning, Grammar, Evolution. *Behavioral and Brain Sciences 26*(6). 651–65.

Kirby, Simon. 1999. *Function, Selection and Innateness: The Emergence of Language Universals*, Oxford University Press.

Kuno, Susumu. 1973. *The Structure of the Japanese Language*, Cambridge, Mass: MIT Press.

Laurence, Stephen & Margolis, Eric. 1999. Concepts and cognitive science. In Eric Margolis & Stephen Laurence (eds.), *Concepts: Core Readings*, 3–82. London, England: MIT press.

McCarthy, John, Paul W. Abrahams, Daniel J. Edwards, Timothy P. Hart & Michael I. Levin. (1985/1962). *LISP 1.5 Programmer's Manual (2nd ed.)*, Cambridge, Mass: MIT Press.

Murphy, Gregory L. 2002. *The Big Book of Concepts*, London, England: The MIT Press.

Pet, Willem. 1987. *Lokono Dian, the Arawak Language of Surniname: A Sketch of its Grammatical Structure and Lexicon*, Cornell University Doctoral Thesis.

Polinsky, Maria. 2012. Headedness, again. *UCLA Working Papers in Linguistics, Theories of Everything 17*(40). 348–359.

Reinhart, Tanya Miriam. 1976. *The Syntactic Domain of Anaphora*, Doctoral dissertation, Cambridge, Massachusets: MIT.

Rips, Lance J. 1995. The Current status of research on concept combination. *Mind and Language 10*. 72–104.

St Clair, Michelle C., Padraic Monaghan, & Michael Ramscar. 2009. Relationships between language structure and language learning: The suffixing preference and grammatical categorization. *Cognitive Science 33*(7). 1317–1329.

Thagard, Paul. 1997. Coherent and creative conceptual combination. In Thomas Ward, Steven M. Smith & Jyotsna Ed Vaid (eds.), *Creative Thought: An Investigation of Conceptual Structures and Processes*, Washington, DC: American Psychological Association.

Tomasello, Michael. 2003. *Constructing a Language: A Usage-based Theory of Language Acquisition*, Harvard University Press.

Tomasello, Michael. 2008. *Origins of Human Communication*, Cambridge, Massachusetts: The MIT Press.

Wisniewski, Edward J. 1997. When concepts combine. *Psychonomic Bulletin and Review 4*. 167–183.

Chris Thornton
University of Sussex
Centre for Research in Cognitive Science
Brighton
BN1 9QJ
UK
c.thornton@sussex.ac.uk