MASTER'S THESIS

LINGUISTICS

Reconsidering Recursion

J. F. van Werven

Under the supervision of

Prof. Dr. M.B.H. Everaert

Utrecht University

January, 2015

And often enough, our faith beforehand in a certain result

is the only thing that makes the result come true.

— William James, *The Will To Believe,* 1897

Contents

Acknowledgments

I would like to express here my gratitude to a number of people thanks to whom I was able to write this master's thesis. First of all, my thanks go out to Martin Everaert, who was willing to be my supervisor and who helped me narrowing down my research questions. We had only a few talks, but they were sufficient to calm me when I believed that linguistics is too big of a scientific chaos to be worth studying. I would also like to thank Eric Reuland for accepting to be my second reader, and in addition Riny Huijbregts, for considering this thesis thoroughly. Your specific comments were much appreciated.

A big thank you goes to the Loekste, for being a fantastic person and for encouraging me to 'diss Chomsky'. I am really lucky to have you in my life. I would also like to thank my dance partner Tobias, who has been very supportive and genuinely interested in the topic of my thesis. Our discussions about embedded music notes, thoughts and sentences were really helpful. Also a word of thanks to Roy, for explaining recursively enumerable sets to me and to Jan-Wouter Zwart, who supported me to investigate recursion in the first place.

I would furthermore like to express gratitude to Maud and Marpessa, for complaining in unison about linguistic matter and alleviating it by having lots of wine and cheese. To Laura, for repeatedly asking and forgetting what recursion is. To Stijn, for arguing about research designs. To Tim, for recognizing recursion in everyday life. To Stephan, for telling me to please, stop investigating recursion. And last but not least, to papa, mama, Rosa, Inge, Fleur, Myrna, Linda, Jesse, Nick, Hester, Anne, Soyeon, Elena and all my lovely classmates for just being there. It is a pleasure knowing you all!

1. Introduction

The so-called recursion only-hypothesis from Hauser *et al.* (2002) has drawn much attention in the field of linguistics. These scholars put forward that the Faculty of Language in the narrow sense „only includes recursion and is the only uniquely human component of the faculty of language" (Hauser *et al.* 2002: 1569). A general problem with this hypothesis is that it does not define precisely enough what it means by recursion. It has been pointed out by a lot of researchers that recursion bears multiple definitions (see, e.g., Parker 2006 and Luuk & Luuk 2011) and it has also been speculated what definition of recursion was most probably meant by Hauser *et al.* (2002) by arguing about its nature and contents (Tomalin 2007; Lobina 2014). However, no research so far focused on possible reasons for the emergence of the recursion confusion by extensively investigating how the term is perceived by different research domains.

The current study therefore investigates to what extent recursion is perceived differently among scientists from different fields. It will be shown that recursion started out as a mathematical term, became a well-known tool in computer science and linguistics and then moved further to psychology and cognitive biology. Due to misinterpretations, the term 'recursion' lost some important characteristics along the way, such as the fact that it is a procedure rather than a structure. It is furthermore investigated to what exactly the different understandings are due by considering the historical background that they draw upon. To this end, chapter 2 focuses on the mathematical definition of recursion and it is shown that the first change in the meaning of recursion happened already when computer scientists used the term in another context. Chomsky was nonetheless well aware of the mathematical nature of recursion he based his Generative Grammar on and therefore used it correctly with respect to mathematics. Chapter 3 investigates the recursion-only hypothesis and shows nevertheless some inconsistencies in Chomsky's work by considering the specific role of

recursion in his frameworks. It is subsequently argued that the treatment of recursion as a structural phenomenon is rather unsurprising. In chapter 4 the recursion confusion becomes even clearer when definitions given in linguistic literature are considered. Notions like nested, tail and structural recursion are discussed here and it is argued that they may not be as important for evolutionary theories about language as is usually thought. Chapter 5 investigates how recursion is seen from a psychologist's and a cognitive biologist's perspective. It appears that recursion is often intuitively linked to notions like theory of mind and solving the Tower of Hanoi, but these are not necessarily the result of a recursive procedure. Furthermore, studies aiming to test for recursion in the auditory, visual and sensory-motor domain are discussed and it is argued that these studies do not succeed in capturing recursive procedures either. Chapter 6 provides an overview of the causes of the emergence of the recursion confusion and elaborates on these findings by reconsidering the implications of the recursion-only hypothesis. Finally, the remainder of the thesis provides some suggestions regarding terminology.

2. Recursion in the formal sciences

2.1 Recursion in mathematics

Originally, recursion was a pure mathematical term. Tomalin (2006: 61) provides a short outline of the history of recursive functions in mathematics wherein he indicates that the basic concept of a recursive function had been introduced in the nineteenth century and that Dedekind deployed such functions when investigating the nature of the real numbers. Dedekind's work influenced Peano, who used such techniques while defining mathematical induction and a number of other mathematicians and logicians developed the theory further.

The basic theory, however, was endowed when Gödel used primitive recursive functions in his incompleteness theorem of 1931. His work was advanced by Church, Kleene, Turing and Post during the 1930s and 1940s. As Chomsky mostly based his ideas on the work of Gödel and Post in particular, it is important to consider the notions of definition by induction and recursively enumerable sets, which are both primarily associated with Gödelian recursive function theory.

2.1.1 Recursive functions

According to Brainerd and Landweber (1974: 54), it is useful to define functions „using some form of induction scheme…, a general scheme… which we call recursion." This way of defining functions is called 'definition by induction', or simply 'recursive definition'. It consists in „defining a function by specifying each of its values in terms of previously defined values" (Cutland 1980). A definition often taken to exemplify this, is that of the factorial class. The factorial class ($fact(n) = n \times n - 1 \times n - 2 \times ..1$) can be recursively defined in the two-equation system that is characteristic of these functions, as follows:

1. If $n = 1$, then $fact(n) = 1$
2. If $n > 1$, then $fact(n) = n \times fact(n - 1)$

The first step is called the base case and the second one is the recursive step which involves another invocation of the factorial function. In other words, in the second step the function calls itself. Thus, if we are to calculate 4!, the definition tells us that since $4 > 1$, we have to calculate $4 \times 3!$. In order to calculate 3! then, we have to calculate $4 \times 3 \times 2!$ and so on, until we reach the base case. From this example we can identify two characteristics of recursion: firstly, its self-reference and secondly, the possibility of defining an infinite set with a finite statement (Tomalin 2006; Tomalin 2007). The latter of course depends on whether a base case is defined or not; if there is no such case, the function will call itself infinitely many times and hence, an infinite set can be defined.

That the factorial class can be computed in a recursive manner does not mean that this is the only option; it can also be computed without calling itself. Lobina (2011: 160) informally describes this alternative way of computing the factorial class as follows: „[the other solution] is simply that factorials can be iteratively computed if we first multiply 1 by 2, then the result by 3, then by 4, until we reach *n*. That is, we keep a running product, together with a counter that counts from 1 up to *n*. Further, we add the stipulation that *n*! is the value of the product when the counter exceeds *n*."

At this point, it is crucial to note that it is true for all (primitive) recursive relations that they can be reduced to recurrence or iterative relations (Rice 1965), in contrast to what a lot of linguists seem to think. For instance, Karlsson (2010: 50) states that full-blown recursion, by which he means nested (and supposedly structural) recursion, is not convertible to iteration.[1] In the same line, Parker (2006: 186) tells us that „only tail recursive algorithms can be re-coded iteratively" and Fitch (2010: 78) claims that iterative functions are inadequate for generating center- and/or self-embedding. It is

[1] I will turn to the issues of nested, tail and structural recursion in chapter 4.

true that not all non-primitive recursive functions can be reformulated as iterative functions, but as natural languages are primitive recursive, we do not need to pay attention to non-primitive recursive functions for the present discussion. The importance of the possibility to convert recursion to iteration may seem questionable at this point, but as we will see, it bears strongly on the claim of Hauser *et al.* (2002).

2.1.2 Recursively enumerable sets

In 1936, Church introduced the idea of a recursively enumerable set and Post influentially expanded this idea. Tomalin (2006: 64) explains what it is about: „a set of non-negative natural numbers, A, is recursively enumerable if there is a general recursive function ϕ which enumerates the members of the set. Thus, the set A is recursively enumerable if $\phi(0) \in A$, $\phi(1) \in A$, $\phi(2) \in A$, ... where ϕ is a general recursive function, and where the sequence constitutes an enumeration of the members of A." Examples of recursively enumerable sets of positive integers are to be found in Post (1944: 285) and a simple one of them is taken over in (1) below.

(1) $\{ 1^2, 2^2, 3^2, \ldots \}$

Post explains this example as follows: „the set [in (1)] is given by a recursive enumeration thereof via the recursive function x^2." This is a bit confusing though; the function x^2 does not appear very 'recursive', as it does not call itself. What he means is that in principle, the algorithm could be fixed in a recursive manner. In regard of this, a simple set like the set of positive integers is also recursively enumerable, simply because each member can be obtained by adding 1 to another (previously defined) number. In contrast, the set of prime numbers is not recursively enumerable, since there is no algorithm that is able to list all of the prime numbers, one by one.

Furthermore, Post frequently used the verb 'to generate' in order to describe how a recursively enumerable set is obtained from its associated general recursive function and it is this term that would give name to Chomsky's Generative Grammar (Tomalin 2007: 1790). Before entering the linguistic field, however, it will be investigated how recursion has been used in computer science, because this formal science has its effects on linguistics as well.

2.2 Recursion in computer science

Soare (1996: 285) defined a computation as „a process whereby we proceed from initially given objects, called inputs, according to a fixed set of rules, called a program, procedure, or algorithm, through a series of steps and arrive at the end of these steps with a final result, called the output." A procedure is much like a mathematical function, except that a procedure must be effective (Lobina 2010: 2). A procedure that calls itself is, unsurprisingly, called a recursive procedure and computer scientists make use of these operations to a great extent. For an example, let us again take the factorial class that was shown in the previous paragraph. When executed, the recursive procedure to compute the factorial of 4 results in the process in (2).

(2) 4 x (fact 3)
4 x (3 x (fact 2))
4 x (3 x (2 x (fact 1)))
4 x (3 x (2 x (1))
4 x (3 x 2)
4 x 6

Lobina (2010: 3) notes that there are two points to be made about these types of processes: first, the process can only end if the operations are carried out in the right order, resulting in a hierarchy among the operations and second, the so-called interpreter is required to keep track of the operations. Turning to the process the iterative procedure that was given executes, we see that no such hierarchy is involved:

(3) (factiter 4 1 1)
(factiter 4 2 1)
(factiter 4 3 2)
(factiter 4 4 6)
(factiter 4 5 24)

In (3), the first digit indicates the number whose factorial is to be computed. The second digit is the counter and the third one is the product, which always starts at 1.

As noted before, all recursive relations can be reduced to recurrence or iterative relations. Computer scientists have also established that any task than can be resolved recursively can also be resolved iteratively (Joyanes Aguilar 2003). However, sometimes one way of resolving is preferred over the other. According to Roberts and Zelenski (2003: 176), recursion is defined as „any solution technique in which large problems are solved by reducing them to smaller problems of the same form." This definition makes clear that a computer scientist is mainly concerned with simplicity – he chooses for recursion, only when the problem is complex enough to require it. Moreover, as Roberts (2006: 8) points out, recursion only 'works' when the subproblems are so simple that they can be solved without further subdivision and it must be possible to combine the results of solving the subproblems into a solution for the original problem. In case of the factorial class we saw above,

these criteria were met. It should however be clear that in itself, nothing about the factorial class is 'recursive-ish'; recursion is simply a handy method when calculating the members of it. Or, as Roberts (2006: 7) puts it: „it is important to recognize that 'recursiveness' is a property of the solution to a problem and not an attribute of the problem itself".

Apart from preferring one way of resolving a task over another (because of for instance memory load or elegance), there is an important difference in the way computer scientists sometimes talk about the internal constitution of the objects computational processes manipulate. That is, although mathematicians restrict the recursive property to functions, rules or sets, computer scientists allow themselves to call objects recursive as well (Lobina 2014: 59).[2] More importantly, this does not only happen when an object is the outcome of a recursive procedure; Rodgers and Black (2004) for instance effectively define a recursive data structure as an object or class „that is partially composed of smaller or simpler instances of the same data structure".

In other words, the word recursion is sometimes used by computer scientists to describe a structure that is obtained by applying a function or rule that is not recursive. This is also clear from the definition Wirth (1983: 149) provides: „an object is called recursive if it contains itself as a part or if it is defined with the aid of itself". For a computer scientist who is well aware of the fact that it is only a matter of convenience to talk in terms of recursive structures and that this term does not necessarily mean that they were generated recursively, this is not at all problematic. Unfortunately, as we will see, this is not clear for everybody.

[2] A set is recursive if and only if both the set and its complement are recursively enumerable (Odifreddi 1989: 140). A recursive set is thus not to be confused with a recursively enumerable set, which was explained on page 10. Moreover, the class of recursively enumerable languages properly includes the class of recursive languages (Hopcroft & Ullman 1969).

2.3 Recursion in linguistics

2.3.1 The influence of mathematics on linguistic thinking

It is probable that certain linguists were aware of the recursive nature of natural language before the 1950s, but as a clear starting point we could take the year that Bar-Hillel published 'On Recursive Definitions in Empirical Science', which was 1953 (Tomalin 2007: 1785). In this paper, Bar-Hillel argued that recursive definitions should also be deployed in the context of empirical sciences, such as linguistics. From the following extract it becomes clear that Chomsky was directly influenced by this: „[a]t one point, Bar-Hillel suggests that recursive definitions may be useful in linguistic theory; whether this turns out to be the case or not, I agree in this instance with the spirit of his remarks (Chomsky 1955: 45).

Also when Chomsky later recalls his early days at MIT, he points out the influence of mathematics: „Mathematical logic, in particular recursive function theory and metamathematics, were becoming more generally accessible, and developments in these areas seemed to provide tools for a more precise study of natural language as well" (Chomsky 1975[1955]:39). Chomsky was thus well-informed about the mathematical background of recursion and did not intuitively make use of it, as so many other scholars nowadays do. That is, Chomsky was acquainted with recursive function theory and he did not obtain his ideas concerning recursively enumerable sets from later expositions of the theory (Tomalin 2007: 1790). In the next section the basic ideas of his framework will be considered.

2.3.2 Generative Grammar

In his dissertation, Chomsky described the goal of syntactic study as follows: „[it is] to show that the complexity of natural languages, which appears superficially to be so formidable, can be analyzed

into simple components; that is, that his complexity is the result of repeated application of principles of sentence construction that are in themselves quite simple" (Chomsky 1975[1955]: 57). To this end, he extensively investigates the construction of grammars and a grammar is „a device for generating sentences" (Chomsky 1975[1955]: 67). There are important assumptions that underlie the construction of grammars, such as the following two (see also Tiede & Stout 2010). Firstly, sentences can be modeled as strings of symbols and a language *L* can be modeled as a set of such strings, either finite or infinite. Secondly, the grammar is a finite device that generates all and only the strings of *L* in a determinate way (Chomsky 1975[1955]: 71).

Focusing on the former assumption first, it should be clear that Chomsky is concerned with the set of sentences that is possible *in theory*. In other words, he is not interested in the actual set of sentences uttered by an individual, as this would be a task of grammatical performance. Instead, he is concerned with competence and by this term he means „the cognitive state that encompasses all those aspects of form and meaning and their relation, including underlying structures that enter into that relation, which are properly assigned to the specific subsystem of the human mind that relates representations of form and meaning" (Chomsky 1980: 59).

Part of the latter assumption in principle already follows from the syntactic goal defined above; if the grammar were to be infinitely great it would be far from simple. As Tiede and Stout (2010: 148) point out, the human linguistic knowledge that is aimed to be modeled is represented in the brain, which is a finite medium as well.

At the heart of his Generative Grammar (henceforth, GG) Chomsky placed a phrase structure component and described it as follows: „[a] phrase-structure grammar is defined by a finite vocabulary (alphabet) V_p, a finite set Σ of initial strings in V_p, and a finite set F of rules of the form: $X \rightarrow Y$, where X and Y are strings in V_p" (Chomsky 1956: 117). As an example of the finite set F, consider (4) ((13) in Chomsky 1957: 26).

(4) (i) *Sentence* → *NP* + *VP*

(ii) *NP* → *T* + *N*

(iii) *VP* → *Verb* + *NP*

(iv) *T* → *the*

(v) *N* → *man*, *ball*, etc.

(vi) *Verb* → *hit*, *took*, etc.

The rules in (4) are called rewrite rules and they occupy an important role in Chomsky's GG. The standard phrase-level conversion *Sentence* → *NP* + *VP* for instance states that a sentence can be rewritten as a noun phrase followed by a verb phrase. Thence, having the initial string *S*, the derivation in (5) can be obtained from *F* ((14) in Chomsky 1957: 26).

(5) *Sentence*

NP + *VP*	(i)
T + *N* + *VP*	(ii)
T + *N* + *Verb* + *NP*	(iii)
The + *N* + *Verb* + *NP*	(iv)
The + *man* + *Verb* + *NP*	(v)
The + *man* + *hit* + *NP*	(vi)
The + *man* + *hit* + *T* + *N*	(ii)
The + *man* + *hit* + *the* + *N*	(iv)
The + *man* + *hit* + *the* + *ball*	(v)

Constructed as such, the grammar indeed takes a very simple form and it is finite. Chomsky (1957: 34-48) is however quick to note the limitations of such a phrase structure description. Apart from other things, this grammar fails to derive complex sentences such as 'I think that the man hit the ball' and as we will see in the upcoming chapters, sentences of this type will raise both great interest and questions.

2.3.3 Capturing infinity

During the 1950s syntacticians began to emphasize that although the grammar of a given natural language must itself be finite, it must also be able to produce a potentially infinite number of sentences (Tomalin 2006: 168). This requirement exists because of what Pullum and Scholz (2010: 115) call 'the no maximal length claim': for any English expression there is another expression that is longer. Indeed, for every well-formed sentence, it is for instance possible to add 'I think that' in front of it. Many scholars have therefore concluded that the (possible) set of grammatical sentences is infinite (see, e.g., Harris 1957; Bach 1964 and Katz 1966).

Chomsky (1959: 137) certainly agrees with this point of view when he states that „[…] any language in which we are likely to be interested is an infinite set". In addition to the no maximal length claim, he takes the creative aspect of language as evidence: „an essential property of language is that it provides the means for expressing indefinitely many thoughts and for reacting appropriately in an indefinite range of new situations" (Chomsky 1965: 6). Recall that at the time, recursive function theory was becoming more accessible, and this perfectly fit the need for a linguistic device that required „infinite use of finite means" as recursion is able to do exactly that: creating infinity by use of finite means.

Other linguists were eager to note this as well. For instance, Bach (1964: 164) puts forward that a grammar needs recursive symbols in order to generate an infinite set of strings and Katz

(1966: 121-122) nicely summarizes that „the creative aspect of language thus provides us with good reason to hypothesize that some of the rules that compose a speaker's knowledge are recursive rules." After all, „these rules are finitely statable, yet they specify an infinite output because they are capable of being endlessly reapplied to their own output to yield unbounded sets of formal objects." At this point it suffices to stress that recursion became a crucial factor in GG for the same reason it became a crucial factor in finitistic proofs. That is, it provided a way of analyzing the potentially infinite in terms of the finite (Tomalin 2006: 168).

3. Recursion as the key to human language

3.1 The recursion-only hypothesis

In 2002, Chomsky wrote an article on the evolution of the Language Faculty, together with Hauser and Fitch (henceforth abbreviated as HCF). The article became famous, especially because of a hypothesis that was made. As the hypothesis concerns recursion, this thesis may not lack an outline of their thoughts and as we will see, HCF's hypothesis made recursion an interesting notion for other research domains as well.

HCF divide two conceptions of the faculty of language, known as FLB (Faculty of Language in the Broad sense) and FLN (Faculty of Language in the Narrow sense) and the latter is included in the former. Apart from FLN, FLB includes at least two other organism-internal systems, called 'sensory-motor' and 'conceptual-intentional'. FLN is described as follows: „[it] is the abstract linguistic computational system alone, independent of the other systems with which it interacts and interfaces" (Hauser *et al.* 2002: 1571). With 'linguistic computational system' they refer to narrow syntax; the system that generates internal representations. Although there is much discussion on the internal architecture of FLN, HCF put forward that „all approaches agree that a core property of FLN is recursion, attributed to narrow syntax".

Unfortunately, the authors do not provide a specific definition of recursion; they only mention that „FLN takes a finite set of elements and yields a potentially infinite array of discrete expressions" and they give the following explanation of the core property of discrete infinity:

> Sentences are built up of discrete units: There are 6-word sentences and 7-word sentences, but no 6.5-word sentences. There is no longest sentence

> (any candidate sentence can be trumped by, for example, embedding it in "Mary thinks that…"), and there is no non-arbitrary upper bound to sentence length. In these respects, language is directly analogous to the natural numbers. (Hauser *et al.* 2002: 1571)

Furthermore, HCF come forward with three hypotheses about the evolution of the Faculty of Language. The third one appeared to be rather controversial, as it gained a lot of critical attention in the literature. The authors hypothesize that most of FLB is based on mechanisms shared with nonhuman animals, whereas FLN is recently evolved and uniquely human (Hauser *et al.* 2002: 1573). And since HCF define FLN as the computational mechanism of recursion, they indeed propose that recursion is unique to our species.

The recursion-only hypothesis means two things. On the one hand, it proposes that recursion is restricted to language. That is, recursion is not to be expected in other cognitive systems. On the other hand, it proposes that recursion is restricted to humans. Thus, recursion is not to be expected in the communication systems of other animals. Of course, in order to elaborate on these thoughts, it is of great importance what exactly they mean by the term recursion.

Recall that the idea behind Post's theory was that all elements belonging to a recursively enumerable set are in the range of a general recursive function. Chomsky (1959: 137) has explained why it is this type of set that is important for the basic generative grammar framework: „Since any language in which we are likely to be interested is an infinite set, we can investigate the structure of *L* only through the study of the finite devices (grammars) which are capable of enumerating its sentences. A grammar of *L* can be regarded as a function whose range is exactly *L*. Such devices have been called 'sentence-generating grammars'."

In this regard, it seems clear that with the term recursion, Chomsky just refers to this and only this property of a grammar. That is, the fact that the finite grammar is able to recursively combine elements, yielding an infinite output regardless of what this output looks like. Nonetheless, the recursion-only hypothesis appeared to be not clear at all for many researchers. Lobina (2014: 62) seems rather astonished by this when he exclaims: „what on earth is there to disagree about?" In the upcoming sections I will however aim to show that the confusion the recursion-only hypothesis revealed, is not very surprising with respect to the many different descriptions and usages of recursion which, partly, exist because of Chomsky himself.

3.2 The proper place for recursion

According to Palmatier (1972: 142), Chomsky has offered two proposals for localizing the recursive property of a grammar. On the one hand, in his 1957 framework, recursion was restricted to the transformational component of the grammar, a component in addition to the phrase structure component that was discussed in chapter 2. On the other hand, in Chomsky's 1965 framework (an improved version of the earlier 1957 framework), the recursive property was moved to the phrase structure component. Chomsky's newest framework, Minimalism, did not exist yet when Palmatier described the role of recursion, but will be discussed here as well.

3.2.1 The 1957 framework

Let us first focus on the initial proposal of the 1957 framework. Recall that the finite set *F* that is part of the phrase structure component contains rules of the form $X \rightarrow Y$. It may be clear that no recursion is present in the phrase structure component and as a result, a phrase generated by this

component cannot be recursive. Thus, a simple sentence like 'I see the dog' is not recursive, because it has not been recursively generated.

In addition to the phrase structure component, there was a transformational component that operated on a given string and converted it into a new string. By means of this, a complex sentence could be derived. As Bickerton (2009: 534) explains: „strings that provided descriptions of simple sentences then served as input to the transformational component. However, for heuristic purposes the operations were frequently described as if they operated on real (surface structure) sentences. Thus 'The man you saw yesterday is Harry's brother' might be described as being produced by insertion of 'You saw the man yesterday' into 'The man is Harry's brother' to yield 'The man [you saw (the man) yesterday] is Harry's brother' with subsequent deletion of the repeated 'the man'."

The insertion Bickerton speaks of is obviously a case of sentential embedding, as it concerns two CP's that are combined. Interestingly, according to Chomsky (1965: 137), this marks the recursive property: „[i]n the earlier [i.e. 1957] version of the theory, the recursive property was assigned to the transformational component, in particular, to the generalized transformations…" From this statement it is clear that sentential embedding was considered to be the result of a generalized transformation, which, in turn, was considered to be a recursive operation.

One might wonder how exactly a transformation can be linked to the notion of recursion, because such an operation is not very reminiscent of, for instance, calculating the factorial class. That is, the complex sentence 'The man you saw yesterday is my brother' is not really defined in terms of itself; it just exists of two parts that are combined by means of the transformational component. However, the fact that the transformational component may operate on a sentence that has been embedded, i.e. transformed, already (cf. Chomsky 1957), enables the grammar to do this an indefinite number of times. In other words, the output of a transformation becomes the input of the

next and thus (part of) the procedure involves running the procedure anew, as befits a recursive procedure.

3.2.2 The 1965 framework

Turning to the proposal of the 1965 framework, Chomsky (1965: 137) asserts that „…the recursive property is [now] a feature of the base component, in particular, of the rules that introduce the initial symbol *S* in designated positions in strings of category symbols." Note, that this is a definition of recursion that differs from the mathematical definition to a rather great extent. According to this, a rewrite rule like $NP \rightarrow NP + S$ is not recursive because the symbol on the left of the arrow appears on the right of the arrow (as the mathematical definition would convey), but because the initial axiom symbol *S* appears on the right of the arrow (Sinha 1978: 173). Hence, just like in the 1957 framework, an embedded sentence is by definition the result of a recursive procedure and therefore signals it. Also, the simple sentence 'I see the dog' is still obtained by a simple non-recursive rule application.

Moreover, Chomsky (1965: 137) states that „[t]here are, apparently, no other recursive rules in the base." However, in *The Logical Structure of Linguistic Theory* Chomsky asserts the opposite:

> When we turn to the level of phrase structure, we find that certain rules may have a recursive character. Thus *Noun Phrase (NP)* might be analyzed in such a way that one of its components may be a *NP*, as in such a sentence as 'the man who made the discovery is my brother', which might be derived by means of such conversion as
>
> $NP \rightarrow NP_1 - \text{who} - VP$
>
> $VP \rightarrow V - NP^3$

> Conversions of this sort will permit generation of infinitely many sentences by that part of the grammar that deals with phrase structure […] (Chomsky 1975[1955]: 171-172)

Defined as such, a recursive rule will always yield a self-embedded structure, e.g. an *NP* in an *NP* or a *VP* in a *VP*. An *NP* in an *NP* cannot be the result of anything else than the rule *NP* → *NP* + …. Therefore, in any event, the embedded structure of a phrase is directly linked to the generating rule.

3.2.3 Minimalism

In Minimalism, rewrite rules are no longer involved. As Zwart (2011: 45) summarizes: „[i]nstead, structure is created by a single operation Merge, which Chomsky (1995) defines as taking two elements and combining them in a set:

(6) *Merge*

i. select x

ii. select y

iii. create $\{x, y\}$

[…] [i]t is (most implicitly) understood that the next time Merge runs, it takes the output of the previous run as part of its input." The elements Merge takes as an input may thus be already formed by earlier Merge operations, which provides an inductive definition of a recursive procedure. This is reminiscent of what the transformational component did, but a crucial difference is that the transformational component only operated on (sets of) strings of symbols, whereas Merge „recursively constructs syntactic objects from items in *N* [i.e. Numeration] and syntactic objects

already formed" (Chomky 1995: 226). Below it will be shown that this difference changes the role of recursion rather drastically.

We saw that according to both the 1957 and the 1965 framework the simple sentence 'I see the dog' was not recursive, because it was obtained by non-recursive rewrite rules, i.e. it had not been generated recursively. In Minimalism, this sentence is thus derived by the repeated application of Merge, which is illustrated in (7).

(7) Given Σ = {I, see, the, dog}

Step 1: Merge (the, dog), resulting in {(the dog), I, see}

Step 2: Merge (the dog, see), resulting in {(see the dog), I}

Step 3: Merge (see the dog, I), resulting in {(I see the dog)}

Since Merge applies recursively and since it applies at every stage of a derivation, even the simple sentence 'I see the dog' is recursively generated. In other words, now a simple sentence is recursive because of the exact same reason why a complex sentence was recursive in the older frameworks; it consists of parts that have been recursively combined, albeit smaller than the combined 'sentential parts' we saw above in Bickerton's example.

3.2.4 Overview

In summary, we have seen different proposals for the proper place of recursion in GG: first it was situated in the transformational component, then it moved to the phrase structure component and it ended up being the sole core of GG. These different places affect the role of recursion and, to some extent, even its meaning. Firstly, in the 1957 framework and also in Minimalism, recursion is present because of the circumstance that elements are combined recursively, whereas in the 1965

framework, recursion is more or less 'preprogrammed' by means of recursive rewrite rules. Secondly, accordingly, sentences are considered to be recursive for different reasons. This is summarized in table 1 below.

Framework	**Simple sentence** e.g. I see the dog	**Complex sentence** e.g. I see that the dog walks
1957	Not recursive	Recursive, because it is derived from other elements
1965	Not recursive	Recursive, either because it involves an embedded *S*, or because it is derived by means of a recursive rule
Minimalism	Recursive, because it is derived from other elements	Recursive, because it is derived from other elements

Table 1 – Recursiveness of a sentence

It might well be the case that mathematicians and Chomsky would not agree with this terminology, as strictly speaking, a sentence may not be called recursive: only a set of sentences (a language) can be recursive. However, due the fact that the output of a recursive function or rule is often called 'recursive', this is how many linguists nowadays assume the term is to be applied.[3] More of these misconceptions will extensively be shown in chapter 4.

3.3 The emergence of a heated debate

The recursion-only hypothesis has intrigued many scholars in the field. It has been contested by e.g. Jackendoff and Pinker (2005a,b), Everett (2005; 2007; 2009) and Parker (2006), but also upheld by

[3] It is generally accepted, for instance, that the Fibonacci sequence may be called a recursive sequence

e.g. Fitch *et al.* (2005), Nevins *et al.* (2009) and Watumull *et al.* (2014). In this paragraph a short overview of criticism and defense is given.

3.3.1 Criticism on the hypothesis

In 2005, Pinker and Jackendoff (PJ, henceforth) wrote direct replies to the HCF article in which they disagree with the recursion-only hypothesis (see Jackendoff & Pinker 2005 and Pinker & Jackendoff 2005). In their articles PJ mostly focus on purely definitional issues, like the proper contents of FLN and FLB. They argue for instance that many more things apart from recursion belong to FLN.

More important for us, however, is the definition of recursion they provide. It is a definition that is not present in the HCF article: „[r]ecursion refers to a procedure that calls itself, or to a constituent that contains a constituent of the same kind" (Pinker & Jackendoff 2005: 203). The first half of the definition is in line with the mathematical definition we saw in section 2.1.1. The second half, however, is reminiscent of the way computer scientists talk about recursion. Recall that they speak of recursive structures independently of how they were generated. The following extract suggests that PJ do this too.

> Recursion consists of embedding a constituent in a constituent of the same type, for example a relative clause inside a relative clause (*a book that was written by the novelist you met last night*), which automatically confers the ability to do so ad libitum (e.g. *a book* [*that was written by the novelist* [*you met on the night* [*that we decided to buy the boat* [*that you liked so much*]]]]). (Pinker & Jackendoff 2005: 211)

One might argue that they are not considering the generation of a relative clause here, but only describing the structure the clause has, thereby mixing up what I will henceforth refer to as

structural recursion and procedural recursion.[4] However, with regard to the previous paragraph, it is clear that in the first two frameworks Chomsky himself linked the notion of sentential embedding to the notion of recursion. That is, according to those frameworks a structure like the one PJ speak of can only be derived by means of a recursive procedure. In this regard, it is not strange at all that PJ assume a relative clause to be an example of recursion.

Similar assumptions are to be found in Everett (2007; 2009). Everett studied the Pirahã language for a long time and claimed that the language lacks recursion. This claim is founded on the fact that, according to him, Pirahã does not have rules of the type in (8) and (9) ((2) and (3) in Everett 2007: 4-5).

(8) a. A → BC

b. B → DE

c. C → AF

(9) A → AB

It may be clear that the recursive (rewrite) rules in (8) and (9) are equal to the rules of the 1965 framework and that apparently, Everett assumes that repeated embedding can only be the result of a recursive procedure.

3.3.2 Defense of the hypothesis

Fitch *et al.* (2005) wrote a response to PJ's critique in order to clarify their proposals. They point out some misunderstandings from PJ and sketch again the distinction between FLN and FLB.

[4] A detailed description and explanation of these terms is to be found in chapter 4.

Unfortunately, in the whole paper they do not address the problematic definition of recursion that PJ provided, thereby (probably accidentally) encouraging their assumption that unlimited nested embedding signals recursion. According to Lobina (to appear), Fitch has in fact a similar definition of recursion in mind (cf. Fitch 2010). That is, he as well refers to self-embedding operations when he discusses recursion. This is rather peculiar – if Chomsky meant to refer to the recursive mechanism Merge, then this was the right moment to make that very clear (at least to Fitch, who was one of the co-authors of the HCF article!).

On the contrary, Everett's confusion of self-embedding structures with recursion has in fact been pointed out by Nevins *et al.* (2009). In a footnote they put forward that Merge is category-neutral and that as a result, „every case of a phrase contained in a larger phrase counts as a demonstration of the rule's recursivity" (Nevins *et al.* 2009: 366). They furthermore argue that earlier models within GG were quite different in this regard, i.e. that the rules for structure-building operations were then category-specific. Indeed, this is in line with what was shown in the previous paragraph. Since Merge applies at every stage of the derivation, every phrase consisting of more than two elements is the output of a recursive procedure. And because Pirahã has phrases that contain more than two words, Nevins *et al.* argue, it is just as recursive as all other languages.

To sum up, both PJ and Everett have been shown to have a definition of recursion in mind that is neither in line with the mathematical definition, nor with Chomsky's definition of Merge. However, as mentioned before, this is not surprising at all when considering Chomsky's own focus on the internal application of recursion within specific rules in his first frameworks. The next chapter will deal with even more definitions of recursion that arose after recursion was introduced in linguistic theory and the difference between procedural and structural recursion will be explained in more depth.

4. Confused by recursion

4.1 Multiple definitions

The term recursion comes with a lot of different definitions, albeit all close connected. In this chapter it will be investigated what associations linguists have when considering recursion and what exactly is problematic about this.[5]

In chapter 2 it was put forward that in linguistics, recursion is used as a tool to capture infinity. At least, this is why Chomsky introduced the term in the first place. Having a look at the literature, recursion is often considered to be only that: a causer of infinity. Take for instance the three definitions given in table 2.

Source	Definition of recursion
Lobeck (2000: 37)	„[r]ecursion expresses the property of language that we can generate phrases of indefinite length"
Carnie (2002: 57)	„[r]ecursivity is the property of loops in the phrase structure rules that allow infinitely long sentences, and explain the creativity of language"
Adger (2003: 20)	„[l]anguages are recursive in nature, allowing linkage between an infinite array of meanings and an infinite array of structures"

Table 2 – Recursion as a causer of infinity

Obviously, it is true that recursion is able to cause infinity, but such a description does not say much about the term itself. It was mentioned before that iteration is able to do so too. Luuk and Luuk

[5] The definitions in the tables of this chapter were taken from Parker (2006).

(2012: 1943) put forward that „[a]ll open-ended sets (e.g. language expressions, *N*) can be defined inductively, i.e. recursively in the broadest sense. For example, one can have the following inductive definition of 'bear': (a) *Ted is a bear*; (b) *All entities that share ate least 98% of Ted's genome are bears*. Observe that, although the set of potential 'bears' is open-ended and inductively defined, no recursion is computationally necessary to determine its contents. An iterative process that compares Ted's genome to that of potential 'bears' would do the job." Thus, the definitions in table 2 fail to recognize the crucial characteristic of recursion: its self-reference.

More specific definitions of recursion can be found in table 3. These definitions have in common that in a recursive procedure, something needs to be 'the same': Christiansen talks about the reappearance of a non-terminal symbol; Lobeck speaks of phrases of the same category; and Pinker less formally talks about phrases of the same kind.

Source	Definition of recursion
Christiansen (1994: 13)	„[r]ecursion entails that the non-terminal symbol on the left-hand side of a rule reappears on the righthand side of the same or another rule"
Lobeck (2000: 36)	„The property of phrase structure rules to generate phrases of the same category"
Pinker (2003: 18)	„[a] phrase is defined as a sequence of phrases, and one or more of those daughter phrases can be of the same kind as the mother phrase"

Table 3 – Recursion as a causer of sameness

At first glance, it seems convenient to talk in terms of a sameness condition when considering recursion. However, this is still neither formal, nor specific enough. As we saw towards the end of the previous chapter, recursivity of a sentence really depends on how clear you are with respect to

defining the categories. Imagine, for instance, a recursive rule stating that in every object with form *X*, an object with the same form is embedded. When taking 'square' for *X*, the rule will yield the structure in figure 1, but it will impossibly yield the structure in figure 2.

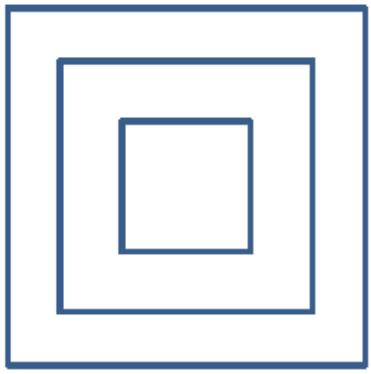

Figure 1 – Square structure

Figure 2 – Geometrical structure

Hence, figure 1 shows a recursive structure and figure 2 does not. However, if we were to assume a less specific form *X*, e.g. 'geometrical', then all of a sudden the structure in figure 2 becomes recursive as well. Therefore, it is not necessary to talk in terms of objects or phrases with the same category when defining recursion.

Table 4 shows a third type of definition of recursion, wherein recursion is seen as some sort of phrase structure. Note that besides the fact that these definitions again speak of categories, there is something else at stake. That is, the first three citations seem to consider recursion to be only some sort of structure (reminiscent of what PJ and Everett did), while Carnie (2002: 22) has a rather procedural meaning in mind.

Source	Definition of recursion
Kirby (2002: 1)	„A property of language with finite lexica and rule-sets in which some constituent of an expression can contain a constituent of the same category"
Newmeyer (2004)	„[r]ecursion is sentences embedded inside of sentences inside of sentences, ad infinitum"
Trask (1993: 229)	„[t]he phenomenon by which a constituent of a sentence dominates another instance of the same syntactic category"
Carnie (2002: 22)	„The ability to embed structures iteratively inside one another"

Table 4 – Recursion as a kind of phrase structure

In the next paragraph it will become clear that all definitions in table 4 fail in differentiating between recursion as a structure and recursion as a procedure and why this distinction is so important.

4.2 Procedural recursion versus structural recursion

One of the most important distinctions that should be made within the meaning of recursion is the one between structural and procedural recursion. Structural recursion was more or less illustrated in table 4 already – it refers to the phenomenon that an object can contain itself, i.e. self-embedding. Procedural recursion, on the other hand, revolves around the fact that some procedure can invoke itself. Note that the first type of recursion defines the make-up of some object, while the latter defines some execution of action. Strictly speaking, only procedural recursion is in line with the mathematical definition as stated in chapter 2.

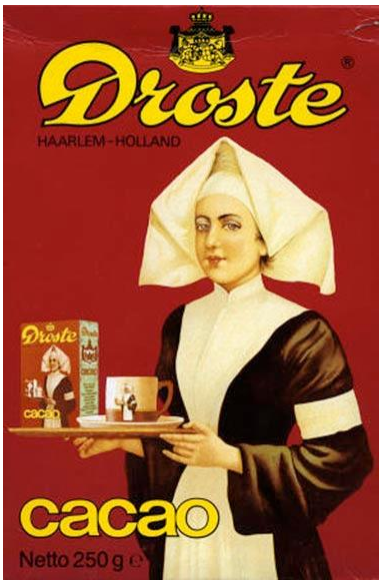

Figure 3 – Droste Cacao

Structural recursion can be illustrated by the following example. When one looks into a mirror that is arranged at a certain angle to another mirror, you do not only see a reflection of yourself in the mirror, but also the reflection of your reflection. That reflection, in turn, contains the reflection of the reflection of your reflection, and so on. This is structural recursion, because each reflection is embedded inside another and relies on the larger reflection existing. Zwart (2011: 43) illustrates structural recursion by giving the famous Droste can picture that can be seen in figure 3. The picture contains an instance of the same picture, i.e. an instance of itself.

In turn, procedural recursion can be illustrated by the example of the way a cake is cut into an even number of pieces (see Kinsella 2010: 180). First one cuts the cake in half, which results in two pieces. Then one cuts each of those pieces in half, resulting in four pieces. Each of those four pieces cut in half again results in eight pieces, and so on. It is procedural recursion because it describes a procedure wherein each piece of cake is defined in terms of the previous one.

Usually, when taking a certain structure into consideration, automatically also a procedure is assumed that could have generated the structure. In case of a self-embedded structure as exemplified in figure 3 above, i.e. a structure that displays structural recursion, the procedure that often is assumed to create this is procedural recursion, obviously because of their (assumed) shared recursive nature. Procedural recursion *can* indeed result in structural recursion. However, this need not be the case. A recursive structure could also have been derived in a non-recursive way. As Zwart (2011: 43) puts it, the Droste can picture could also have been generated by assigning a color to each pixel, which is in fact more likely. This way of creating the picture clearly has no link to recursion

whatsoever. Moreover, procedural recursion does not necessarily result in structural recursion either. The cake cutting example was an instance of procedural recursion and yet, the sliced cake does not show a recursive structure. That is, there is no way of regarding the pieces of cake as being embedded in themselves or one another.

Indeed, Luuk and Luuk (2012) put forward that recursion (i.e. procedural recursion) and self-embedding (i.e. structural recursion) are logically independent and also Lobina (2014: 64) strongly emphasizes that „the obvious point to make […] is that the existence of recursive structures […] does not necessarily mean that they were, or indeed, that they must be, generated recursively."

4.3 Nested recursion and tail recursion

Parker (2006) puts forward that most definitions fail to differentiate between two other types of recursion as well, known as nested recursion and tail recursion. She argues that the importance of the difference „bear strongly on the claim of Hauser *et al.* (2002), and in a larger context on theories of the evolution of language" (Parker 2006: 172). In this paragraph the notions of nested recursion and tail recursion will be discussed and it will be argued that the distinction may not be as important as Parker suggests.

4.3.1 Nested recursion

Nested recursion is also called center-embedding. This means that the embedding occurs in the middle of a sentence, phrase or other object. A sentence often taken to exemplify nested recursion is shown in (10).

(10) [The mouse [the cat [the dog chased] bit] ran]

Note that (10) is an instance of structural recursion; it displays the make-up of a sentence rather than describing the process of producing it. It can be seen that material exists on both sides of the embedded component. The phrase 'the dog chased' is embedded in the larger phrase 'the cat bit' and in turn, that phrase is embedded in 'the mouse ran'. Another example of nested recursion was the embedding of mirrors that we saw in the previous paragraph.

According to Kinsella (2010: 180), an important property of nested recursion is that it involves keeping track or adding to memory using a pushdown stack. In paragraph 2.2 the notion of keeping track was shortly mentioned as well, but it is important to realize that this is only of importance when performing an actual execution of a recursive function or procedure. This is correctly captured in the definition of recursion that is provided by Liu and Stoller (1999: 73): „[r]ecursion refers to computations where the execution of a function or procedure calls itself and proceeds in a stack fashion."

In computer programming, a stack is used in order to hold data in such a way that access to that data is restricted. This stack is a last-in-first-out type of storage device; only the last element pushed onto the stack, can be popped off again to be retrieved. Turning to the sentence in (10), one might indeed need a pushdown stack, but only when *parsing* the sentence. Parsing is the process of recognizing an utterance or string, rather than the process of generating it.

When parsing (10), first the NP 'the mouse' is encountered. Reaching the next word, it becomes clear that 'the mouse' cannot be retrieved yet because there is another NP 'the cat'. As Parker (2006: 176) summarizes: on entering the recursive call, we have to store the first NP somewhere so that when we reach the final verb 'ran' we can associate it with. Indeed, the

pushdown stack then, is the place to store the NPs until a verb is found to associate it with. It is nicely illustrated by Hofstadter how difficult this parsing process can be:

> The grammatical structure of all languages involves setting up quite elaborate push-down stacks, though, to be sure, the difficulty of understanding a sentence increases sharply with the number of pushes onto the stack. The proverbial German phenomenon of the "verb-at-the-end", about which droll tales of absentminded professors who would begin a sentence, ramble on for an entire lecture, and then finish up by rattling off a string of verbs by which their audience, for whom the stack had long since lost its coherence, would be totally nonplussed, are told, is an excellent example of linguistic pushing and popping. (Hofstadter 1979: 151)

Nevertheless, one might argue that parsing a sentence, i.e. the procedure to accept a sentence (belonging to the language *L*), need in itself not be recursive; it depends on whether or not the grammar that one uses in order to parse the sentence might be called recursive. Thus, both the very long sentence in this extract and the sentence in (10) are very complex with respect to their structure, but again, the complexity of (parsing) the structure does not imply similar complexity of the procedure that generated it.

4.3.2 Tail recursion

Opposed to nested recursion, tail recursion occurs at the beginning or ending of a phrase or sentence, shown in respectively (11) and (12) below.

(11) [[[[John's] father's] friend's] cat] is playing

(12) [The girl that kissed [the guy that saw [the man that ate [an apple]]]]

In (11) and (12), there is only material on one side of the embedded component, either on the left or on the right. Tail recursion does not require that we keep track while parsing the sentence and there are only local dependencies. A third type of sentence, at first glance similar to tail recursion, can be seen in (13) below. Note that (12) and (13) contain the exact same NP's. The structure of the sentence is, however, rather different.

(13) [The girl] and [the guy] and [the man] ate [an apple]

The difference in structure between (12) and (13) becomes clear when some of the NP's switch places. Switching 'the guy' and 'the man' gives a whole other meaning to the sentence in (12), which is shown in (14). Doing the same thing in (13) does however not change the meaning at all (see (15)).

(14) [The girl that kissed [the man that saw [the guy that ate [an apple]]]]

(15) [The girl] and [the man] and [the guy] ate [an apple]

Such a constituency test reveals that (13) and (15) have a rather flat (iterative) structure, whereas (11), (12) and (14) have a more hierarchical ('recursive') one.

It was shortly mentioned before that many linguists assume that only tail recursion is convertible to iteration (see Karlsson 2010: 50 and Fitch 2010: 78). This is indeed the reason why

Parker (2006: 172) assumes that the distinction between nested recursion and tail recursion is important for the recursion-only hypothesis; if a nested recursive structure could only be the result of a recursive procedure and not of an iterative one, it would make sense to investigate whether there exist languages that lack nested recursive structures. It should now however be clear that the distinction between nested and tail recursion is only useful in the sense that it enables us to investigate sentence structures and phrase distributions and that it does not tell us anything about generating procedures and underlying mechanisms.

5. Recursion in performance

5.1 An intuitive approach of recursion

Ever since HCF's article was published, other scientific domains than linguistics started to reconsider recursion as well. This paragraph discusses the ideas of Michael Corballis, who is one of the psychologists that work on recursion. Similar ideas of some other researches are discussed too and it will be argued that they all treat recursion fairly intuitively and incorrectly with respect to its historical mathematical background.

5.1.1 Theory of mind

Corballis has written multiple articles about recursion and even a whole book, called *The recursive mind* (Corballis 2011). Remarkably, when defining recursion in the very first chapter of this book, Corballis does not refer to any mathematical source. Instead, he refers to the dubious definition of PJ that we saw on page 23. Moreover, he states that „[t]he second part of [PJ's] definition is important, especially in language, because it allows that recursive constructions need not involve the embedding of the same constituents, […], but may contain constituents of the same kind – a process sometimes known as 'self-similar embedding'" (Corballis 2011: 6). Recall however that it was exactly this part of PJ's definition that was problematic for two reasons: it allows recursion to be some sort of structure instead of a procedure and it makes use of the word 'kind', which, as was shown in 4.1, is too vague.

Although Corballis uses language phenomena in his book, the main theme is that recursion originates in thought rather than in language. This idea is based on another citation of PJ: „[t]he only reason language needs to be recursive is because its function is to express recursive thoughts. If

there were not any recursive thoughts, the means of expression would not need recursion either" (Pinker & Jackendoff 2005: 230). By recursive thought, Corballis means theory of mind: the ability to imagine what might be going on in the mind of another individual. This, he argues, is recursive because it involves „the insertion of what you believe to be someone else's state of mind into your own" (Corballis 2011: 133). It suggests that according to him, a simple embedding does not involve recursion. Indeed, in an ealier paper Corballis (2007: 244) states that „[t]he mental processes of thinking, knowing, perceiving or feeling might be regarded as zero-order theory of mind, and are probably common to many species. They are not recursive." Thus, Corballis assumes that recursion is only implied by higher order theory of mind, which refers to thinking, knowing, perceiving or feeling what others are thinking, knowing, perceiving or feeling.

Corballis is not the first to link recursion to the notion of theory of mind. For instance, Takano and Arita (2006: 405) as well argue that if we assume that individuals with theory of mind also consider others and themselves to have theory of mind, then there „should be a recursive structure here." They illustrate this idea with the picture in figure 4.

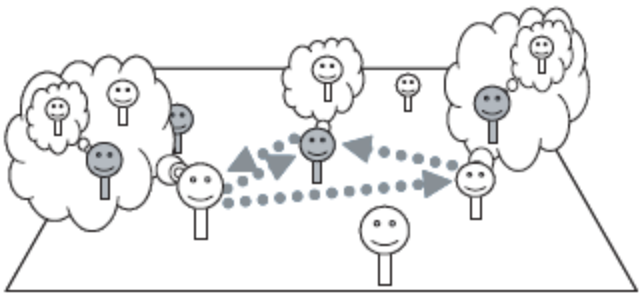

Figure 4 – 'Recursive thinking with a theory of mind'

It is clear that just like so many other researchers, Corballis, Takano and Arita directly link a self-embedded structure to recursion, without really considering the procedure that creates it.

5.1.2 Recursive problem solving

Corballis takes tool making as another example of human's 'recursive skills'. It is well-known that many animals are capable of using tools; chimpanzees for instance use stones to crack nuts and crows use twigs to extract insects from holes (Corballis 2007: 247). But human beings are believed to be the only animals that use tools in order to make other tools and according to Corballis, this again implies recursion.

Partially similar reasoning is to be found in Schiemenz (2002). At first glance it seems that Schiemenz as well bears a structural definition of recursion in mind when he argues that recursive objects are surrounding us, thereby referring to pieces of music, language and pictures that contain smaller, but similar pieces of themselves. However, he furthermore states that one can solve problems recursively if they contain problems of the same class, which is a rather procedural definition of recursion. As an example of recursive problem solving he mentions the insertion of cards into a stack. We might understand this as follows: if you were to alphabetically insert a card named 'Crefcoeur' into a stack of other name cards, you start out by solving the subproblem as to where a card named 'C' should go. The next step involves solving the exact same problem, yet now you have to find the right place for 'R' within the stack of C cards you just found and so on, until you have found the proper place for the card.

Another example of recursive problem solving is a technique one might use when solving the famous Tower of Hanoi puzzle (see, e.g., Goel & Grafman 1995). This puzzle was invented by the French mathematician Edouard Lucas and it involves three pegs and some disks on one of them. As the rules specify, the disks on each peg are stacked so that smaller disks are always on top of the larger disks and the goal is to move all the disks to another peg. In order to fulfill this goal, one could use a recursive strategy because just like in the insertion of cards example, the puzzle can be divided into similar 'subpuzzles' until a subpuzzle is encountered that is easy enough to solve

right away. Recall the recursive way a factorial can be computed; when resolving the Tower of Hanoi this easiest subpuzzle would represent the base case.

5.1.3 No account of mental representations

Intuitively, all of the above examples seem to have something to do with recursion. Although Corballis does not offer an account of how the mental representations of theory of mind are formed, one might argue that recursive thinking can really only be the result of the recursive procedure of embedding thoughts in one another. That is, it may in theory be possible to yield an embedded construction expressing embedded thoughts by executing an iterative procedure, but why would you assume such an inelegant solution? This is however exactly the problem; we do not know what kind of solution the brain prefers. Even though it may intuitively seem right to speak of recursive insertion of thoughts in one another, there is no evidence that this is what happens behind the scenes.

As for the procedures of inserting a card into a stack and resolving the Tower of Hanoi: they can be well described recursively, but neither are these procedures evidence of some recursive mechanism. Resolving the Tower of Hanoi as described above does not necessarily mean that a person needs recursion for this. You might as well just try moving pegs and simply 'happen' to resolve it.[6] To summarize, having theory of mind, making tools in order to make tools and other procedures that are often believed to reveal a recursive mechanism do not tell us much more than that human beings entertain and make use of recursive mental representations.

[6] See Goel & Grafman (1995) for four different strategies in resolving the Tower of Hanoi.

5.2 Probing recursion

5.2.1 Geometrical fractals

A scientist who is well aware of the fact that brain computations are opaque to observers is Maurício Martins: he argues that we cannot posit any underlying mechanism until behavioral correlates have been found (Martins 2012: 2055). Therefore, he proposes a new paradigm to test for recursion; a paradigm that is said to be able to assess the ability of participants to extract a recursive rule. In order to test for recursion in the visuo-spatial domain, he makes use of the Visual Recursion Task (henceforth, VRT). This task is based on geometrical fractals, which are figures that exhibit a repeating pattern displaying at every scale. An example of a well-known fractal is given in figure 5.

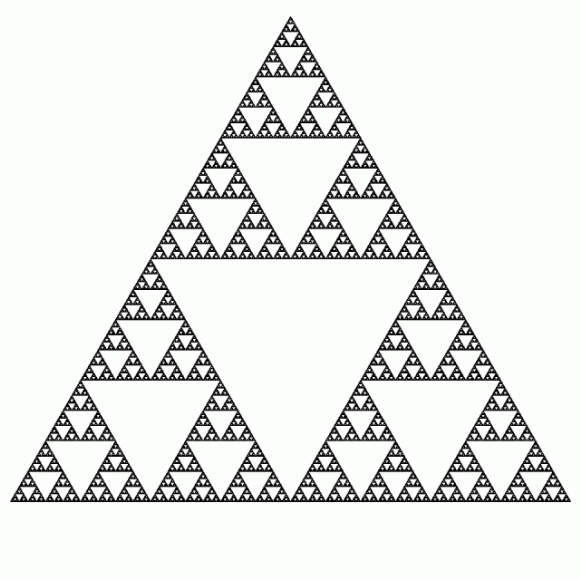

Figure 5 – Sierpinski triangle

In the VRT, subjects get to see the first three steps of a fractal generation procedure and are then asked to select the next right step out of two possible candidates (see figure 6). The first step shows

a single triangle, the second step shows the addition of three smaller triangles and a third step adds some more. Being able to select the right candidate is then said to mark the ability to extract a recursive rule. As a control task there is an Embedded Iteration Task (see figure 7). In this task, „iterative processes embed constituents within fixed hierarchical levels, without generating new levels" (Martins 2012: 2065).

However, having regard of figure 6, we might ask ourselves whether we can really speak of hierarchical levels here. That is, why would we assume that the smaller triangles popping up from step 2 onwards depend on a larger triangle existing? Let us for instance compare the fractals to embedded sentences. If in a sentence a large CP is to be deleted, everything inside it is gone as well. For the triangles this does not hold. The smaller triangles happen to appear at the edge of larger tri-

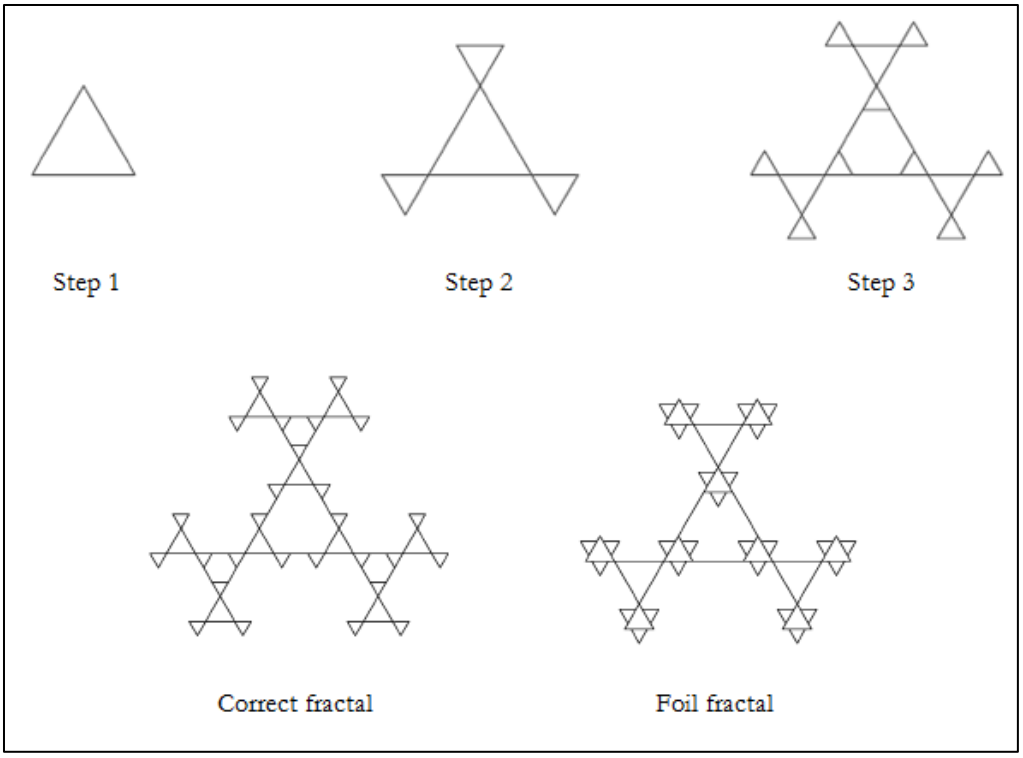

Figure 6 – Visual Recursion Task

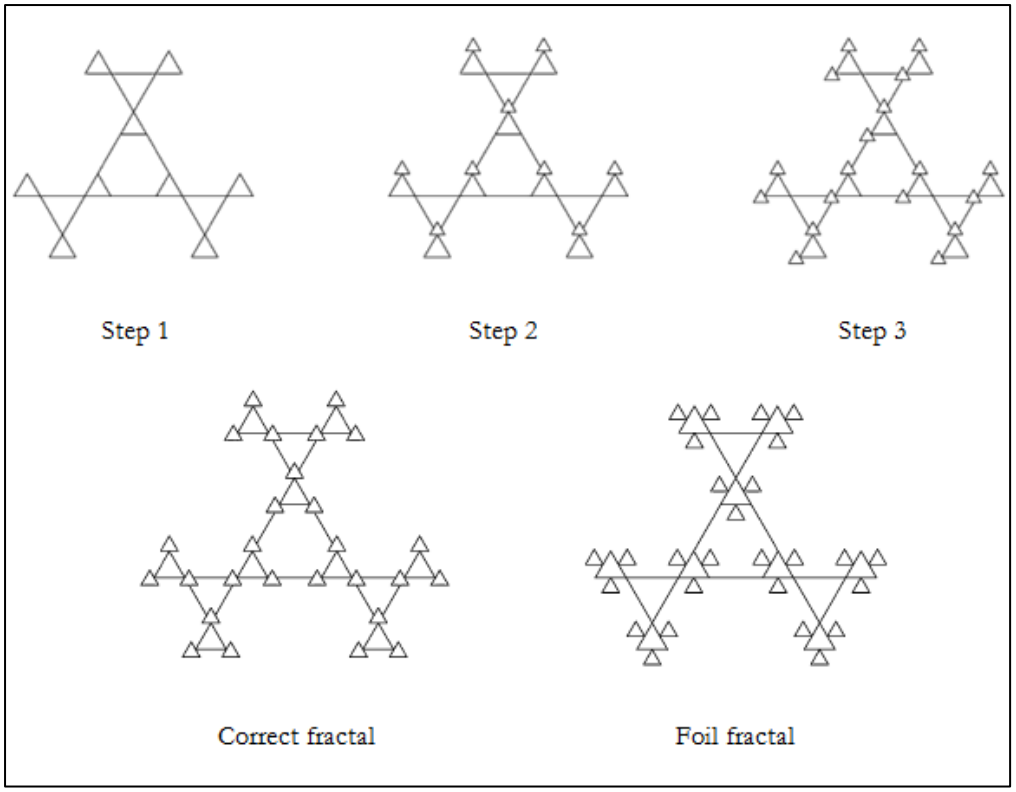

Figure 7 – Embedded Iteration Task

angles, but this does not mean that if a large triangle were to disappear, these smaller triangles would follow. For a more thorough review of Martins work, see Lobina (to appear: 7-10) who marks this dubious point as well and furthermore emphasizes again that a self-embedding rule is not to be confused with a recursive rule, because the self-embedding property is an aspect of *what* a rule does, but not of *how* it proceeds (his emphasis).

5.2.2 Tone hierarchies

In similar tasks, Martins (2014) aims to test for recursion in the auditory domain. To this end the Auditory Recursion Task (ART) and Auditory Iteration Task (AIT) are developed. In these tasks tone hierarchies are used instead of visuo-spatial coordinates, as exemplified in figure 8 below.

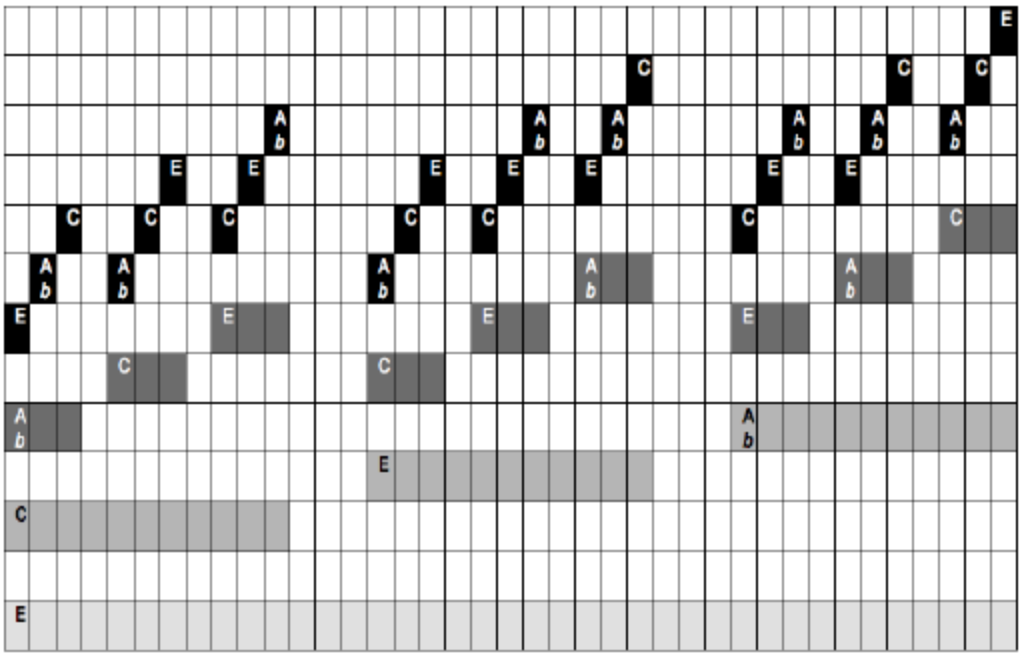

Figure 8 – Auditory Task with tone hierarchies (Martins 2014)

As Martins (2014) explains: E, C and A*b* are musical notes and each rectangle represents one of them. The horizontal axis represents the tone duration and the vertical axis represents pitch. Pitch and duration of each triad of contiguous tones is determined by the pitch and duration of the subjacent tone in the hierarchy. The hierarchical levels are denoted by different colors and in the ART they were presented cumulatively, with the lower levels presented first. The procedure of generating these 'musical fractals' thus parallels the procedure described for visuo-spatial recursion in 5.2.1.

Musical fractals have been studied before. Henderson-Sellers & Cooper (1993) have for instance investigated the proposal that classical music has a fractal nature. They put forward that „if a 'musical landscape' were fractal, we should see this as we moved from the low resolution piece (a small number of notes) to the more complete work. At each additional step of resolution self-

similarity would tell us to expect identical musical landscapes, but in miniature" (Henderson-Sellers & Cooper 1993: 282). As an example they provide the composition given in figure 9 on the next page and explain it as follows: in (a), there is a randomly chosen four note phrase; in (b) more detail added using crotchets with the underlying semibreve theme retained; in (c) a second recursion to semiquavers and in (d) a further recursion to semi-demi-semiquavers. Interestingly, they thus make use of the word recursion to describe what happens in the process of 'fractally deriving' the composition. However, it is said nowhere in the article that this is the only way such a composition can be derived, nor is it claimed that one 'needs' or 'shows' recursion when perceiving the composition as being self-similar, as Martins seems to suggest.

Moreover, when we zoom in on his Auditory Task, it can be seen that opposed to the composition in figure 9, the first, lowest layer does not display any structure; it is simply a long during tone. We might ask ourselves in what way the second layer, i.e. the slightly darker grey layer that starts with a C note, is structurally similar to this lowest layer then. Who or what assures us that a participant listening to these sequences of tones perceives them as being hierarchically self-similar and therefore picks out the right candidate? It is possible for a participant to pick out the right candidate without this exact perception and since self-similarity is not even a (mathematical) requirement of a fractal (see Dodge & Bahn 1986: 191), it seems that Martins is neither investigating distinctive signatures of recursion, nor real fractal structures.

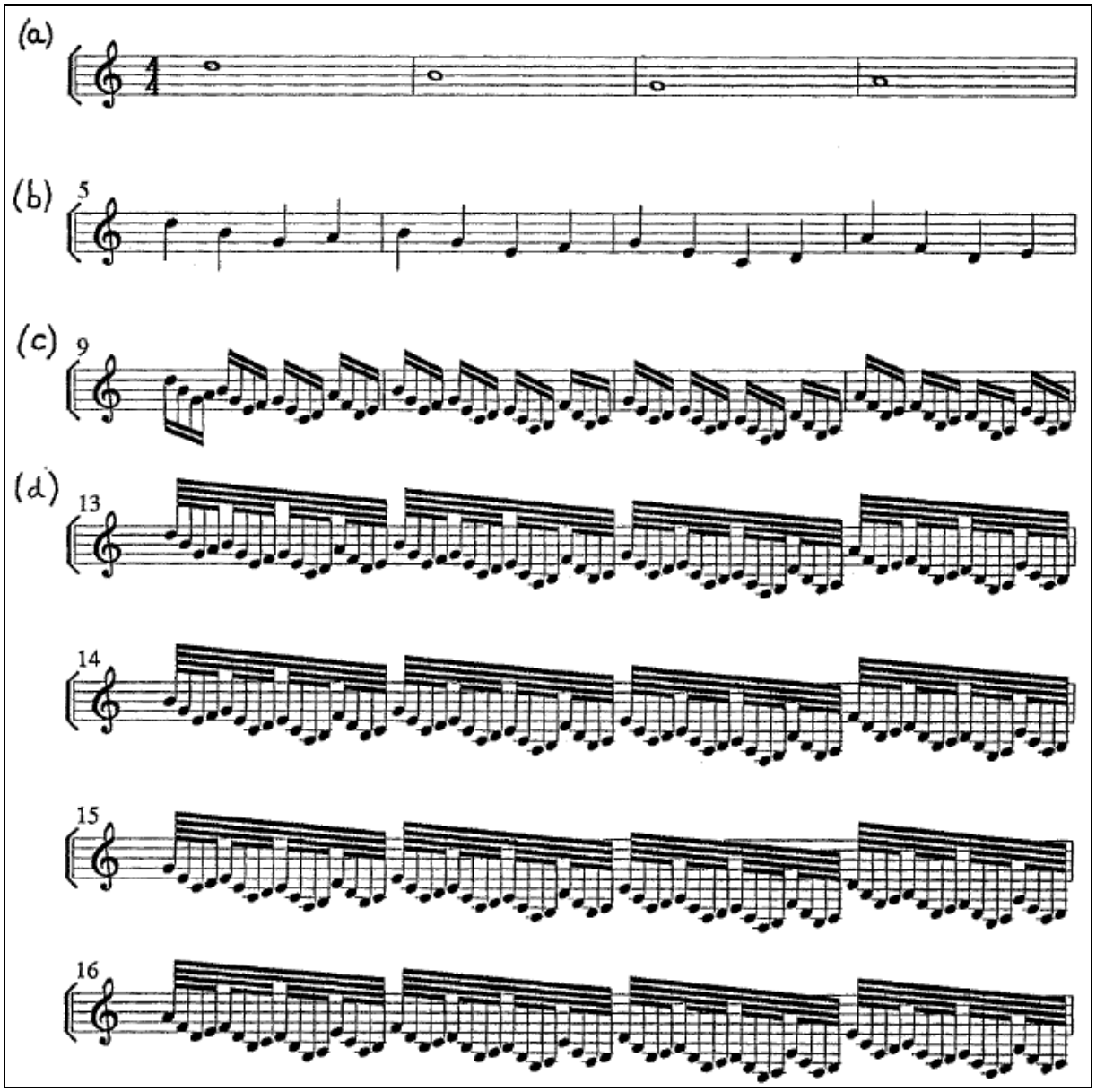

Figure 9 – Fractally derived composition

5.3 To describe is not to explain

In the previous paragraphs it was shown that there seems to be a tendency among scholars to assume that if a procedure can be described in a recursive manner, then it is indeed recursion that is responsible for the procedure to happen. Even now that a lot of articles have been written about what recursion is and what it definitely is not, there are still researchers who come up with remarkable 'evidence' against the recursion-only hypothesis.

In a very recent paper, Vicari and Adenzato (2014) for instance claim that there is evidence of recursive mechanisms in the structure of intentional action.[7] First of all, they enumerate four central properties of recursion, one of which, they suppose, is „self-embedding: a recursive structure embeds a constituent inside a constituent of the same kind, a recursive process is one that calls itself while the procedure is running" (Vicari & Adenzato 2014: 173). It may be clear that they mix up some properties of procedural recursion and structural recursion here. Later on, Vicari and Adenzato (2014: 275) put forward the hypothesis that „John R. Searle's (1983) philosophical analysis of the intentionality of acting and perceiving can be extended to capture the presence of the crucial features of recursion (self-embedding, long-distance dependence, identity preservation, discrete infinity) in sensory-motor processing."

I will not discuss Searle's philosophical analysis of intentionality here, nor will I focus on the four supposed crucial features of recursion. The crucial point to be made is the following: to extend a certain analysis to capture the presence of features of a certain procedure does definitely not serve as evidence that it is indeed this procedure operating. In short, what Vicari and Adenzato argue for, is that the content of an intention-in-action has a causally self-referential structure specifying the causal relationship between the state itself and its conditions of satisfaction. This means that the

[7] If this were indeed true, it would serve as evidence against the recursion-only hypothesis because sensory-motor processing falls outside the range of FLN; see chapter 3.

content of an intention cannot be as in (16) but should at least be as in (17) (cf. Vicari & Adenzato 2014: 176).

(16) Intention (the arm goes up)

(17) Intention (the arm goes up because of this intention)

They continue to argue that „[t]he content of the intention is, then, causally self-referential, and causal self-referentiality is a self-embedding structure: the content of the state refers to the state itself as causing its own conditions of satisfaction." Of course, this is a nice description of an intended action, but what does it tell us about the underlying mechanism? Even when all features of recursion are recognized in intention-in-action, it would still not mean that the mental operations at stake are indeed operating recursively in performance. In the end, Vicari and Adenzato may thus succeed in extending Searle's analysis to capture the four features of recursion they distinguish, but that is really all they do – they do not provide evidence of any recursive mechanism.

At this point the question arises as to whether it is possible at all to probe recursion. Regarding Roberts (2006) (see also paragraph 2.2), there are three criteria for a recursive solution to be applicable:

1. It should be possible to reduce a complex problem into simpler but architecturally equivalent subproblems.
2. Each of these subproblems should be solvable in the same manner.
3. The combination of all solutions should provide a solution for the entire task.

It thus seems plausible that when probing recursion, at least the above criteria should be met. Moreover, as Lobina (to appear: 6) stipulates: „[w]hat we need to isolate, in any case, is a specific cognitive task for which both a recursive and an iterative solution could be postulated, so that an independent variable – working memory being the crucial variable – could be manipulated in order to work out which process is in fact being executed."

6. Reconsidering recursion

6.1 Three reasons for misunderstandings

In regard of the previous chapters, I indicate at least three reasons for the confusion concerning recursion. First of all, one obvious cause of confusion is the shift in meaning that emerged when recursion entered the computer science. It was shown in chapter 2 that for a mathematician, only a function that appeals to itself may be called recursive, whereas computer scientists also make use of this term to refer to data structures that are partially composed of smaller or simpler instances of the same data structure. This is not strange at all; in fact, it often happens that we describe the make-up of constructions by using a term that strictly speaking refers to an (associated) procedure. Consider for instance logicians who speak of the embedding of sets in other sets or doctors who talk about embedded organs in the body. The noun 'embedding' is clearly derived from the verb 'to embed', but in these examples only the make-up of the object is described. That is, 'embedding' is used as a synonym for 'being inside' and it does not necessarily refer to any procedure or movement. The same holds for calling a structure recursive without actually considering the generating procedure. As long as people are aware of the difference, there is nothing problematic about it. But when procedural recursion and structural recursion are taken to be dependent of each other, misunderstandings arise.

Secondly, in chapter 3 it was shown that Chomsky himself has been rather unclear with respect to using the term recursion in different contexts. Lobina (2014: 59) points out that Chomsky and Miller (1963: 284) explicitly recognize that their computational formalism is underlain by recursion when „they state that the → relation mediating the conversion of some structure ϕ_1, ... ϕ_n

into some structure ϕ_{n+1} can be interpreted as 'expressing the fact that if our process of recursive specification generates the structures ϕ_1, ... ϕ_n, then it also generates the structure ϕ_{n+1}.'" Recall that Hauser *et al.* (2002: 1571) also put forward that just like there is no largest number, there is no largest sentence and language is therefore much like the natural numbers. However, it is confusing that Chomsky (1957[1955]: 171-172) too emphasizes the recursive character of rewrite rules and even asserts that „conversions of this sort [e.g. $NP \rightarrow NP + VP$] will permit generation of infinitely many sentences by that part of the grammar that deals with phrase structure". The latter suggests that generation of infinitely many sentences is possible thanks to the fact that a phrase may contain a phrase of the same category and not so much because of the circumstance that all kinds of syntactic objects are recursively combined. It might well be the case that this particular example was merely accidental and that it was Chomsky's intention to talk about recursion more generally, comprising both indirect and direct recursion. However, pertaining to the way that it was stated, it is not surprising that nowadays many researchers still assume that self-embedded structures signal recursion.

As a third cause of the recursion confusion we might blame the common intuitive understanding of recursion. In chapter 5 a few approaches of recursion were discussed that all show how easily some researchers expect the notion of recursion can be grasped. When considered critically, it appears fairly naïve to think that recursion can be recognized in performance just like that. It is nonetheless striking that researchers like Corballis find themselves adopting a claim that is almost equal to the recursion-only hypothesis: „I propose that recursion is a ubiquitous property of the human mind and possibly the principal characteristic that distinguishes our species from all other creatures on the planet" (Corballis 2007: 240). Just like HCF, Corballis thus proposes that only humans possess a recursive property, but it seems that his claim is founded on a rather different definition of recursion than the claim of HCF (or at least Chomsky) is founded on.

To sum up, recursion started out as a pure mathematical notion bearing a clear definition (but see Soare 1996 for an extensive discussion of what he calls the Recursion Convention), and got a somewhat broader meaning when it entered the computer science. It then became a well-known tool in Chomsky's GG but was misunderstood by many linguists, mostly because recursive procedures were confused with recursive structures and embedding procedures. Along the same line of thoughts, psychologists and cognitive biologists have adopted the idea that the occurrence of self-embedded structures in other domains than language could serve as evidence against the recursion-only hypothesis.

6.2 The recursion-only hypothesis revisited

Now that we are aware of the many misconceptions of recursion, where does that leave us? First of all, it is necessary to reconsider the implications of the recursion-only hypothesis, and a good attempt would be to determine which type of recursion it implicates.

In chapter 2 it was shown that Chomsky was well-aware of the mathematical background of the notion he was going to introduce in linguistic theory. He mostly drew upon Post's production systems and even used the same terminology as becomes clear from statements like the following: „in general, a set of rules that recursively define an infinite set of objects may be said to generate this set" (Chomsky 2006: 112). It was explained that Post was not concerned with self-referential functions per se, but merely with sets being generated in a linear sequence, each new element being effectively obtained from elements previously generated. In all likelihood, it was because of this particular property of a (general) recursive set that Chomsky decided to place a recursive component at the heart of GG. That is, recursion became the distinguishing property of generative grammar,

transformational or otherwise (Tomalin 2007; Lobina 2014, to appear). It seems reasonable to argue subsequently that it is also this recursive property that HCF refer to in their claim, instead of to self-embedding rules or mechanisms.[8]

What would it mean if this were true? Firstly, according to the recursion-only hypothesis, it would mean that only in the linguistic domain, elements are to be recursively combined and secondly, it would mean that only humans are capable of entertaining this exact linguistic recursive property. It would moreover raise the question as to whether another computational mechanism could also suffice to account for the discrete infinity of language, such as an iterative mechanism. Regarding the latter, Luuk and Luuk (2011) have indeed proposed that iteration could do the job as well. However, according to Lobina (2014: 68), Luuk and Luuk do not recognize the recursive nature of linguistic generation because although they correctly note that Merge reapplies its embedding operation in an iterative manner, they fail to realize that every stage of a derivation is stipulated to be recursively defined. In a 'Post-ian' manner, that is. Indeed, nowhere in their article, do Luuk and Luuk (2011) refer to Post or show familiarity with his work and it may thereby be possible that their disagreement with the HCF claim is also simply a matter of disagreement about terminology.[9] The latter being the case would be rather unfortunate because, as is nicely summarized by Soare (1996: 29), „ambiguous and little recognized terms and imprecise thinking lead to poor communication both within the subject and to outsiders, which leads to isolation and a lack of progress within the subject, since progress in science depends upon the collaboration of many minds."

Turning back to the implications of the recursion-only hypothesis, it remains a challenge for future research to investigate whether it is true or not. More precisely, it is a challenge to empirically

[8] Although, as was mentioned before, Fitch presumably has another definition of recursion in mind than Chomsky (see Fitch 2010).
[9] Luuk & Luuk (2011) furthermore disagree because they think natural language is a finite set, in contrast to what HCF assume.

test for recursion and even if recursion were to be found in non-linguistic domains, generative linguists following Chomsky on this point might argue that this non-linguistic recursion is still language-dependent or that the supposed counterexamples concern performance, rather than competence (Vicari & Adenzato 2014: 175). However, it seems that when HCF started to speak of FLN having evolved for reasons other than language and of computations outside the domain of communication such as navigation and social relations (Hauser *et al.* 2002: 1571), Chomsky shifted his perspective of viewing recursion as a merely formal explanatory property of competence to a perspective that allows recursion to be a property of performance. This is remarkable, since Chomsky has always been concerned with competence only. For future work, it is therefore of great importance that we at least carefully distinguish between recursion as a formal explanation and recursion as something that is implemented in the brain because these two are very different things.

6.3 Conclusive remarks

In this study it was my aim to investigate how differently recursion is perceived by several research domains. It is clear that there exist many (mis)understandings of the term, ranging from being strictly formal to fairly intuitive. As a result, at least seven different concepts are called 'recursive' in the literature, as is summarized in table 5 on the next page.

To say the least, the usage of different words denoting these concepts would be welcome. For linguistic theory, Tomalin (2007: 1799) has proposed to replace the word recursion by 'inductive definition', since, as he argues, „if this interpretation of the term is adopted, the 'recursive' components of GG are able to accomplish all that is required of them (i.e., the generation of infinite

structures using finite means) and the correspondence between the human linguistic and arithmetical cognitive functions is not undermined."[10]

		Is recursive, if…
Mathematicians	A function	It calls itself
Post	A set of integers	It is recursively enumerated
Chomsky I	A set of grammatical sentences	It is recursively enumerated
Chomsky II	A rewrite rule	It introduces the initial symbol *S*
Computer scientists	A data structure	It is composed of smaller instances of itself
Linguists	A phrase	It contains a phrase of the same kind
Cognitive biologists	Any (mental) representation	It can be described in terms of itself

Table 5 – Scientist's usage of the word 'recursive'

Ultimately, it would be convenient if the word 'recursive' would either refer to a procedure or to the outcome of a recursive procedure and not to both. This may however be hard to achieve in retrospect, as it is generally accepted that 'recursive structures' refers to self-embedded constructions. Nonetheless, it would be a good attempt to at least always mention whether the kind of recursion being considered is either procedural or structural in order to avoid confusing definitions like the ones that were given in chapter 4. After all, if recursion really is the most fundamental component of the faculty of language, then it surely is of great importance to determine its properties as precisely as possible.

[10] Some contra-arguments are however discussed in Lobina (2010).

References

Adger, D. (2003). *Core Syntax: A Minimalist Approach*. Oxford: University Press.

Bach, E.W. (1964). *An introduction to transformation*. New York: Holt, Rinehart and Winston.

Bickerton, D. (2009). Recursion: Core of complexity or artifact of analysis? In T. Givón & M. Shibatani (Eds.), *Syntactic complexity: Diachrony, acquisition, neurocognition, evolution* (pp. 531–544). Amsterdam: John Benjaminst.

Brainerd, W.S., & Landweber, L.H. (1974). *Theory of computation*. New York: Wiley.

Carnie, A. (2002). *Syntax: A Generative Introduction*. Oxford: Blackwell.

Chomsky, N. (1955). Logical syntax and semantics: their linguistic relevance. *Language* 31, 36–45.

Chomsky, N. (1956). Three models for the description of language. *I.R.E. Transactions on Information Theory* 2(3), 113–123.

Chomsky, N. (1957). *Syntactic Structures*. The Hague: Mouton.

Chomsky, N. (1959). On certain formal properties of grammars. *Information and Control* 2, 137–167.

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

Chomsky, N. (1975[1955]). *The logical structure of linguistic theory*. Cambridge: MIT Press.

Chomsky, N. (1980). *Rules and Representations*. Oxford: Blackwell.

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

Chomsky, N. (2006). *Language and Mind*. Cambridge: Cambridge University Press.

Chomsky, N., & Miller, G.A. (1963). Introduction to the formal analysis of natural languages. In R.D. Luce, R.R. Bush & E. Galanter (Eds.), *Handbook of Mathematical Psychology*, vol. 2 (pp. 269–322). New York: John Wiley.

Christiansen, M.H. (1994). *Infinite Languages, Finite Minds: Connectionism, Learning and Linguistic Structure*. University of Edinburgh dissertation.

Corballis, M.C. (2007). The uniqueness of human recursive thinking. *American Scientist* 95, 242–250.

Corballis, M.C. (2011). *The recursive mind*. Princeton: Princeton University Press.

Cutland, N. (1980). *Computability: an introduction to recursion function theory*. Cambridge: Cambridge University Press.

Dodges, C., & Bahn, C.R. (1986). Musical fractals. *Byte* II(6), 185–196.

Everett, D. (2005). Cultural constraints on grammar and cognition in Pirahã: Another look at the design features of human language. *Current Anthropology* 46, 620–646.

Everett, D. (2007). Cultural constraints on grammar in Pirahã: A Reply to Nevins, Pesetsky, and Rodrigues. *LingBuzz*, April 2007.

Everett, D. (2009). Pirahã culture and grammar: A response to some criticisms.*Language* 85, 405–442.

Fitch, W. T., Hauser, M. D., & Chomsky, N. (2005). The evolution of the language faculty: clarifications and implications. *Cognition* 97(2). 179–210.

Fitch, W.T. (2010). Three meanings of recursion: key distinctions for bioloinguistics. In R. Larson, V. Déprez & H. Yamakido (Eds.), *The evolution of human language* (pp. 73–90). Cambridge: Cambridge University Press.

Harris, Z. (1957). Co-occurrence and transformation in linguistic structure. *Language* 33, 283–340.

Hauser, M., Chomskky, N., & Fitch, W.T. (2002). The faculty of language: What is it, who has it, and how did it evolve? *Science* 298, 1569–1579.

Henderson-Sellers, B., & Cooper, D. (1993). Has classical music a fractal nature? – A reanalysis. *Computers and the Humanities* 27, 277–284.

Hofstadter, D.R. (1979). *Gödel, Escher, Bach: An Eternal Golden Braid*. London: Penguin.

Hopcroft, J.E., & Ullman, J.D. (1969). Formal Languages and Their Relation to Automata. Addison-Wesley.

Jackendoff, R., & Pinker, S. (2005). The nature of the language faculty and its implications for evolution of language. *Cognition* 97, 211–225.

Joyanes Aguilar, L. (2003). *Fundamentos de Programación*. Madrid: McGraw-Hill.

Karlsson, F. (2010). Syntactic recursion and iteration. In H. van der Hulst (Ed.), *Recursion and human language* (pp. 43-67). Berlin/New York: De Gruyter Mouton.

Katz, J.J. (1966). *The philosophy of language*. New York: Harper and Row.

Kinsella, A.R. (2010). Was recursion the key step in the evolution of the human language faculty? In H. van der Hulst (Ed.), *Recursion and human language* (pp. 179-191). Berlin/New York: De Gruyter Mouton.

Kirby, S. (2002). Learning, bottlenecks and the evolution of recursive syntax. In T. Briscoe (Ed.), *Linguistic Evolution through Language Acquisition: Formal and Computational Models* (pp. 173–203). Cambridge: Cambridge University Press.

Liu, Y.A., & Stoller, S.D. (1999). From recursion and iteration: what are the optimizations? *SIGPLAN Notices* 34, 73–82.

Lobeck, A.C. (2000). *Discovering Grammar: An Introduction to English Sentence Structure*. New York: Oxford University Press.

Lobina, D.J. (2010). Recursion and Linguistics: an addendum to Marcus Tomalin's Reconsidering Recursion in Syntactic Theory. *Interlingüística XX*.

Lobina, D.J. (2011). "A running back"; and forth: a review of recursion and human language. *Biolinguistics* 5, 151–169.

Lobina, D.J. (2014). What linguists are talking about when talking about.. *Language Sciences* 45, 56–70.

Lobina, D.J. (to appear). Probing recursion. *Cognitive processing*. DOI 10.1007/s10339-014-0619-z.

Luuk, E., Luuk, H., (2011). The redundancy of recursion and infinity for natural language. *Cognitive Processing* 12(1), 1–11.

Martins, M.D. (2012). Specific signatures of recursion. *Philosophical Transactions of the Royal Society B* 367, 2055–2064.

Martins, M.D. (2014). Recursion is not language domain-specific: interim results of a research program. Paper presented at Evolang X, Vienna. Proceedings of the 10th International conference.

Nevins, A., Pesetsky, D., & Rodrigues, C. (2009). Pirahã exceptionality: a reassessment. *Language* 85. 355–404.

Newmeyer, F. (2004). Cognitive and functional factors in the evolution of grammar. *European Review* 12, 245–264.

Odifreddi, P. (1989). *Classical Recursion Theory. The Theory of Functions and Sets of Natural Numbers.* Amsterdam: Elsevier.

Palmatier, R. A. (1972). *A glossary for English transformational grammar*. New York: Appleton-Century-Crofts.

Parker, A. (2006). Evolution as a Constraint on Theories of Syntax: The Case against Minimalism. PhD thesis, Theoretical and Applied Linguistics. University of Edinburgh.

Pinker, S. (2003). Language as an adaptation to the cognitive niche. In M. Christiansen & S. Kirby (Eds.), *Language Evolution* (pp. 16–37). Oxford: Oxford University Press.

Pinker, S., & Jackendoff, R. (2005). The faculty of language: What's special about it? *Cognition* 95, 201–236.

Post, E. (1944). Recursively enumerable sets of positive integers and their decision problems. In M. Davis (Ed.), *The undecidable* (pp. 304–337). New York: Dover Publications Inc.

Pullum, G., & Scholz, B.C. (2010). Recursion and the infinitude claim. In H. van der Hulst (Ed.), *Recursion and human language* (pp. 113-138). Berlin/New York: De Gruyter Mouton.

Rice, G. (1965). Recursion and iteration. *Communications of the ACM* 8, 114–115.

Rodgers, P., & Black, P.E. (2004). Recursive data structure. In V. Pieterse & P.E. Black, P.E. (Eds.), *Dictionary of Algorithms and Data Structures*. Available at: http://www.nist.gov/dads/HTML/recursivstrc.html.

Roberts, E.S. (2006). *Thinking recursively with Java*. Hoboken: John Wiley and Sons, Inc.

Roberts, E.S., & J. Zelenski. (2003). *Programming Abstractions in C++*. Course Reader, available at: http://www.cas.mcmaster.ca/~qiao/courses/cs2so3/textbook/ProgAbs.pdf

Schiemenz, B. (2002). *Managing complexity with recursion*. Paper presented at the Cybernetics and Systems, Vienna.

Searle, J.R. (1983). *Intentionality*. Cambridge: Cambridge University Press.

Sinha, A.C. (1978). On the status of recursive rules in transformational grammar.*Lingua* 44, 169–218.

Soare, R. (1996). Computability and recursion. *Bulletin of Symbolic Logic* 2(3), 284–321.

Takano, M., & Arita, T. (2006). Asymmetry between Even and Odd Levels of Recursion in a Theory of Mind. *Proceedings of Alife* X, 405–411.

Tiede, H.J., & Stout, L.N. (2012). Recursion, infinity, and modeling. In H. van der Hulst (Ed.), Recursion and human language (pp. 147-158). Berlin/New York: De Gruyter Mouton.

Tomalin, M. (2006). *Linguistics and the formal sciences*. Cambridge: Cambridge University Press.

Tomalin, M. (2007). Reconsidering recursion in syntactic theory. *Lingua* 117, 1784–1800.

Trask, R.L. (1993). *A Dictionary of Grammatical Terms in Linguistics*. London: Routledge.

Vicari, G., & Adenzato, M. (2014). Is recursion language-specific? Evidence of recursive mechanisms in the structure of intentional action. *Consciousness and Cognition* 26, 169–188.

Watumull, J., Hauser, M.D., Roberts, I.G., & Hornstein, N. (2014). On recursion. *Frontiers in Psychology* 4, 1–7.

Wirth, N. (1983). *Algorithmen und Datenstrukturen*. Stuttgart: Teubner Verlag.

Zwart, J.W. (2011). Recursion in Language: A layered-derivation approach. *Biolinguistiscs* 5, 43–56.