TOWARDS A BRAIN COMPATIBLE THEORY OF SYNTAX BASED ON LOCAL TESTABILITY

STEFANO CRESPI REGHIZZI AND VALENTINO BRAITENBERG

Abstract. Chomsky's theory of syntax came after criticism of probabilistic associative models of word order in sentences. Immediate constituent structures are plausible but their description by generative grammars has met with difficulties. The type 2 (context-free) grammars account for constituent structure, but already trespass the mathematical capacity required by language, because they generate unnatural mathematical sets: a consequence of being based on recursive function theory. Abstract associative models investigated by formal language theoreticians (Schutzenberger, McNaughton, Papert, Brzozowsky, Simon) are known as locally testable models. A combination of locally testable and constituent structure models is proposed under the name of Associative Language Description, arguing that it equals type 2 grammars in explanatory adequacy, yet is compatible with brain models. Two versions of ALD are exemplified and discussed: one based on modulation, the other on pattern rules. A sketch of brain organization in terms of cell assemblies and synfire chains concludes.

1. Introduction

Chomsky's theory of syntax came after his criticism of probabilistic associative models of word order in sentences, but the inadequacy of probabilistic left-to-right models (Markov process) had already been noticed by Lashley [12] who anticipates Chomsky's arguments [5] by observing that probabilities between adjacent words in a sentence have little relation to grammaticality of the string. Yet associative models on one hand provide an intuitively appealing explanation of many linguistic regularities, on the other they are aligned with current views on information processing in the cortex. A classical argument pro syntax is that the choice of an element is determined by a much earlier element to be remembered across gaps filled by intervening clauses (constituents). Ambiguity too provides a strong indication that sentences carry a structure. The established model for immediate constituents analysis relies on the Context-Free (CF) grammars. Their rules assign names (nonterminal symbols) to different kinds of constituents (syntax classes). Such grammars cannot handle many non-elementary aspects of language, yet already trespass the mathematical capacity required by language, because they generate unnatural mathematical sets. In our opinion this is a clear indication that this model is misdirected. The related models of greater complexity, such as context sensitive grammars, are even more subject to the same criticism: for instance they generate

Key words and phrases. context-free grammar, word association, cell assemblies, synfire chains, non-counting property, grammar inference.

This work was presented at the *Workshop on Interdisciplinary approaches to a new understanding of cognition and consciousness*, Villa Vigoni, Menaggio 1997. We acknowledge the support of Forschung für anwendungsorientierte Wissenverarbeitung, Ulm and of CNR-CESTIA.

such mathematical languages as the set of strings whose length is a prime number [21], a consequence of being based on recursive function theory.

On this ground and motivated by the search for a linguistic theory that would be more consistent with the findings of brain science, we propose a new model, called associative language description (ALD). This grafts onto the old associative models the immediate constituent structure. The associative theories we build upon have been investigated in the 60's by mathematicians (notably Brzozowsky, McNaughton, Papert, Schutzenberger, Simon, Zalcstein) and are known as locally testable (LT) models. In sect. 2 we recall the LT definitions and we present the ALD models. The plural indicates that ALD is a sort of general approach that can be realised in different ways. The original ALD model exploits LT for specifying both the structure of constituents and their permitted contexts; the bounded ALD model of [17] is a mathematically simpler version that has been used for proving formal properties and for technical applications to programming languages [20]. Then we briefly compare ALD versus CF models, and we show that ALD are easier to infer from examples of structured sentences. In sect. 3 we sketch a brain organization for ALD processing in terms of neural mechanisms. In the conclusion we refer to early related research and discuss possible avenues for continuation.

2. Associative language descriptions

Local testability. Certain frequent patterns of language can be described by considering pairs of items that occur one next to the other in a sentence. The precise nature of the items depends on the level of language description. The items are the phonemes at the phonological level, they are lexemes or word categories at the syntactical level. In our discussion we leave unspecified the level of language description, since what we are proposing is an abstract model of grammar, that in future development will have to be instanciated to account for various linguistic phenomena. The items will accordingly be represented as characters from a terminal alphabet $\Sigma = \{a, b, \dots\}$.

Looking at a string such as $x = abccbcccc$ we note that x contains the following substrings of length 2: ab, bc, cc, cb to be called *digrams* or *2-grams*; we notice that ab, the prefix of the string, is the initial digram; similarly cc is the final digram or suffix. The study of the digrams (or more generally the k-grams, $k \geq 2$) that appear in the phrases has been suggested many times as a technique for characterizing to some extent the grammatically valid strings. In particular, the limits of a Markovian model based on the relative frequency of k-grams occurring in the language corpus, have been examined by Chomsky and Miller [6]. Checking for the presence of certain k-grams in a given string to be analysed is a simple operation for a computer, and one that could very easily be performed by cortical structures, as already noticed by Wickelgren [22], who shows that serial order can be enforced by small associative memories. The recognition algorithm needs a finite memory to be used as a sliding window capable of storing k characters. The window is initially positioned at the left edge of the string to be analyzed. Its content is checked against the set of permitted k-grams. If it matches, the window is advanced by one position on the string, and the same check is performed, until the window reaches the right edge of the string. If the check is passed in all positions of the window, the string is accepted, otherwise it is rejected. Because the sliding window essentially performs a series of local inspections, the languages thus defined have

been aptly called *locally testable* (LT). As this algorithm uses a finite memory, independently of the length of the string, its discriminatory power cannot exceed the one of a finite-state automaton. Not all finite-state languages are LT, but this loss of generative capacity primarily concerns certain periodic patterns, that are irrelevant for modeling human languages. One example would be a string over the alphabet a, b containing any number of a's and a number of b's that is a multiple of 3. Such a property cannot be checked by local inspections over the string. The formal properties of LT languages have been investigated by mathematicians using algebraic and automata-theoretical approaches. An early comprehensive reference is the book on non-counting languages by McNaughton and Papert [13]. Several variations on the notion of LT have been proposed by theoreticians (e.g. see [14]), but we stick to the simplest definition, since nothing is gained, by considering more refined models, at this early stage of our investigation.

We use a special character $\perp$, not present in the terminal alphabet, called the *terminator*, which encloses the sentences of the language.

For $k \geq 1$ a k-gram x is either a string containing exactly k characters of Σ or a possibly shorter string starting or ending by the terminator. Formally: $x \in \Sigma^k \cup \perp(\Sigma \cup \varepsilon)^k \cup (\Sigma \cup \varepsilon)^k \perp$

Definition 1. For a string x of length greater than k, we consider three sets: $\alpha_k(x)$ the *initial k-gram* of x; $\gamma_k(x)$ the *final k-gram* of x; and, if x is longer than $k+1$, $\beta_k(x)$, the set of *internal k-grams* (those that occur in any position other than on the left and right edges). (Notice that internal k-grams may not contain terminators). A *locally testable description* (*LTD*) of order $k \geq 1$ consists of three sets of k-grams: A (initial), B (internal), C (final). A LTD $D = (A, B, C)$ defines a formal language, denoted $L(D)$, by the next condition. A string x is in $L(D)$ if, and only if, its initial, internal, and final k-grams are resp. included in A, B, and C. More precisely, the condition is: $x \in L$ iff, $\alpha_k(x) \in A \wedge \beta_k(x) \in B \wedge \gamma_k(x) \subseteq C$.

As a consequence if two strings x and y have the same sets α, β, and γ either both are valid sentences of $L(D)$ or neither one is. It is often convenient to avoid specifying the exact width of the k-grams. A language L is called *locally testable* if there exists a finite integer k such that L admits a LTD of order k.

From local testability to structure definition. It is obvious that a LTD falls short of the capacity required by natural language, as it permits to define only some finite-state languages. Its major weakness is the inability to define constituent structures, a necessary feature of any syntax model. A simple way to introduce constituent structures into a LTD is now proposed. A constituent has to be considered as a single item, encoded by a new terminal symbol Δ called a *place holder* (PH). By this expedient a constituent containing other constituents gives rise to k-grams containing the PH.

Next, we define the tree structures that are relevant for ALD. A tree (see fig. 1) whose internal nodes are labeled by Δ and whose leaves are labeled by characters of Σ is called a *stencil tree*.

A tree is composed by juxtaposed subtrees having height one and leaves with labels in $\Sigma \cup \{\Delta\}$, called *constituents*. The frontier of a stencil tree T is denoted by $\tau(T)$, while $\tau(K_i)$ denotes the frontier of the constituent K_i.

Definition 2. For an internal node i of a stencil tree T, labeled by Δ, let K_i and T_i be resp. the constituent and the maximal subtree of T having root i. Introduce a

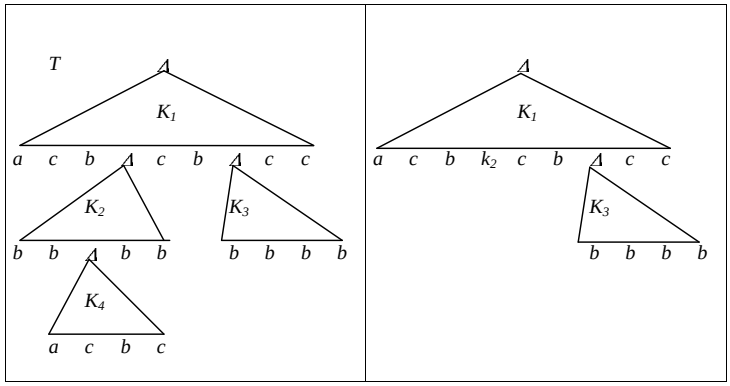

FIGURE 1. A stencil tree T with four constituents schematised as triangles. To the right a condensed tree.

new terminal symbol k_i, and consider the 'condensed' tree $T_{K'i}$ obtained replacing in T the subtree T_i with k_i. Consider the frontier $\tau(T_{K'i})$ of the condensed tree, which can be written as the concatenation of three parts: $\tau(T'_K i) = sk_i t$, where s and t are possibly empty terminal strings. The strings $\perp s$ and $t \perp$ are called, resp., the *left / right context* of the constituent K_i (or of the subtree T_i) in T: $left(K_i, T) = left(T_i, T) = \perp s$ $right(K_i, T) = right(T_i, T) = t \perp$.

For instance, in Fig. 1 the left context of K_3 is *acbbbacbcbbcb* and the right context of K_1 is $\perp$: notice that terminators are automatically prefixed/appended.

ALD based on local testability. The original idea of Associative Language Descriptions (ALD) was entirely founded on the concepts of LT. Each class of constituent is associated to a LT description, i.e. a triple $\langle A, B, C\rangle$, with the provision that k-grams contain place holders, if they contain nested constituents. For graduality we start with a simpler model, where any constituent can freely occur anywhere a PH occurs.

Definition 3. A *free* ALD F consists of a collection $\{D_1, D_2, \ldots, D_m\}$ of locally testable descriptions of order k, each one of the form $D_i = \langle A_i, B_i, C_i\rangle$, where each one of the three symbols represents a set of k-grams possibly containing PHs. Each triple D_i is called a *syntax class*.

A free ALD $F = \{D_1, D_2, \ldots, D_m\}$ defines a formal language, $L(F)$, consisting of a set of stencil trees, by the following condition. A tree T is in $L(F)$ iff for each constituent K of T the frontier $\tau(T_K)$ belongs to the language defined by some syntax class D_i of F. Formally:

$$T \in L(F) \iff \forall K \in T : (\exists D_i \in F : (\tau(T_K) \in L(D_i)))$$

Example 1. Take the alfabet $\Sigma = \{a, +, \times\}$ of certain arithmetic expressions, where a stands for some numeric value. Sums, e.g. $a + a + a$, are defined by the LTD $\langle A = \{a+\}, B = \{a+, +a\}, C = \{+a\}\rangle$ and products are similarly defined by $\langle A = \{a\times\}, B = \{a\times, \times a\}, C = \{\times a\}\rangle$.

Now we extend the language allowing any variable to be replaced by any expression as constituent, thus giving rise to a tree language, which is defined by the free ALD $F = \langle D_1, D_2\rangle$ where

$D_1 = \langle A_1 = \{a+, \Delta+\}, B_1 = \{a+, \Delta+, +a, +\Delta\}, C_1 = \{+a, +\Delta\}\rangle$,
$D_2 = \langle A_2 = \{a\times, \Delta\times\}, B_2 = \{a\times, \Delta \times \times a, \times\Delta\}, C_2 = \{\times a, \times\Delta\}\rangle$.

For instance the stencil tree $\underline{a + a} \times \underline{a + a + a} \times a$, where the constituents are underscored, is valid since the triples α, β, γ exhibited by each constituent are included in D_1 or in D_2. On the contrary the tree $a + \underline{a \times a + a}$ is not valid since the underscored constituent yields $\langle\{a\times\}, \{\times a, a+\}, \{+a\}\rangle$, not included either in D_1 or in D_2.

The license to replace any constituent by another one would cause obvious over generalisation in a grammar, say, of English. For instance if a preposition clause (*in the garden*) occurs in a certain position (*the man in the garden*), then any other clause (e.g. the noun clause *the sharp knife*) would be permitted in the same position, causing acceptance of the ungrammatical string *the man the sharp knife*. In other words all syntax classes of a free ALD are equivalent with respect to their context of occurrence. This insufficient discriminatory capacity is remedied by the next development. The improvement consists in adding, to each syntax class, the indication of the valid contexts of occurrence. Two different manners of so doing have been considered. In the original proposal the contexts are specified by listing the k-grams that may occur at the left and right edge of a constituent. The second manner, introduced in [17], uses patterns to specify at once a syntax class and its permitted contexts.

Definition 4. An ALD G *with modulation*[1] consists of a free ALD

$$F = \{D_1, D_2, \ldots, D_m\}, m \geq 1$$

with the following additions. For each LT description D_i two non-empty sets of k-grams without PHs[2] are given, the *opening* (or left) *passage* L_i and the *closing* (or right) *passage* R_i. Therefore G is defined by a finite collection of components:

$$G = \{E_1, E_2, \ldots, E_m\}$$

where $E_i = \langle D_i, L_i, R_i\rangle$ and $D_i = \langle A_i, B_i, C_i\rangle$. As before each component E_i defines a syntax class.

Before presenting the acceptance condition, we need to formalise the k-grams occurring on the borders of a constituent.

Definition 5. The *left border* $\lambda(K, T)$ of a constituent K of stencil tree T is the set of k-grams, with no PH, such that: they occur in the frontier $\tau(T)$ of T, and they partially overlap the left edge of $\tau(K)$. Reformulating more precisely the second condition, after segmenting the string $\tau(T)$ as $\tau(T) = u\tau(K)v$, we have:

[1] In music 'modulation' indicates a passage leading from one section of a piece to another.

[2] The reason for not allowing PHs is procedural. In order to parse a string, passages must be recognised, which would be inconvenient if the PH, itself the result of a parse, had to be detected in advance. This choice is however one of several possible variations.

$\lambda(K,T) = \{x|(x$ is a k-gram over $\Sigma \cup \perp) \wedge (x = x'x$", where x' is a suffix of $\perp u) \wedge (x$" is a prefix of $\tau(K)v\perp)\}$. The right border $\rho(T,K)$ of a constituent is symmetrically defined.

As an example, from fig. 1 we have the following borders for $k = 2$:

$$\lambda(K_2,T) = \{bb\}, \rho(K_2,T) = \{bc\}, \lambda(K_1,T) = \{\perp a\}, \rho(K_1,T) = \{c\perp\}$$

For $k = 4$: $\lambda(K_3,T) = \{bcbb, cbbb, bbbb\}; \rho(K_3,T) = \{\ bbbc, bbcc, bcc\perp\}$

Intuitively for a stencil tree to be valid each constituent must be preannounced by the occurrence of some k-grams belonging to the opening passage of its syntax class; a symmetrical condition applies to the closing passage.

Example 2. To illustrate the discriminative capacity of modulation, we change Ex. 2, by permitting one containment only: multiplicative expression inside additive expressions. By inspecting a few instances of stencil trees one easily comes out with the ALD:

$A_1 = \{a+, \Delta+\}, B_1 = \{a+, \Delta+, +a, +\Delta\}, C_1 = \{+a, +\Delta\},\ L_1 = \{\perp a\}, R_1 = \{a\perp\}$.

$A_2 = \{a\times\}, B_2 = \{a\times, \times a\}, C_2 = \{\times a\},\ L_2 = \{a + a, +a\times, \perp a\times\}, R_2 = \{\times a+, a + a, \times a\perp\}$.

Notice that at least 3-grams are required to make modulation effective for L_2 and R_2, but for brevity 2-grams have been used in all other sets.

As a test one can verify that the tree $a \times \underline{a + a}$ is rejected because $\times a+$ is not present in the left passage L_1. On the contrary the tree $a + \underline{a \times a}$ is accepted since the left border $\{a+a, +a\times\}$ of the constituent is included in L_2 and its right border $\{\times a\perp\}$ is included in R_2.

Bounded ALD. Recently the original definition has been reshaped into a mathematically simpler model [17], which is more directly comparable with CF grammars, yet it does not decrease capacity. The model will be here called *bounded ALD* to distinguish from ALD with modulation.

In a sense bounded ALD stay to ALD with modulation in a similar relation as CF grammars stay to CF grammars with regular expressions: stencil trees of bounded ALD are bounded in degree, because recursion is used instead of iteration for producing repetitive structures. The other difference between the two models has to do with the manner of specifying permitted contexts.

Definition 6. A *bounded* ALD A consists of a finite collection of rules of the form $\langle x, y, z\rangle$, usually written $x[y]z$, where $x \in (\perp\Sigma*) \cup \Sigma^+$, $y \in (\Sigma * \perp) \cup \Sigma^+$, and $z \in (\Sigma \cup \{\Delta\})*$. For a rule the string z is called the *pattern* and the strings x and y are called the *permissible left/right contexts.* If a left/right permissible context is irrelevant, it can be replaced by the "don't care" symbol $'-'$.

A tree T is *valid* for a bounded ALD A iff for each constituent K_i of T there exists a rule $u[\tau(K_i)]v$ where u is a suffix of the left context of K_i in T and v is a prefix of the right context of K_i in T.

The language $L(A)$ defined by A is the set of stencil trees valid for A.

Sometimes we also consider the set of strings corresponding to the frontier of a tree language. This allows us to talk of the *string language* defined by a ALD.

Example 3 The string language $\{a^n cb^n | n \geq 1\}$ is defined by the rules:
$\perp[a\Delta b]\perp, a[a\Delta b]b, a[c]b$. Both contexts can be dropped from the first two rules, and

the right (or left but not both) context can be omitted from the last rule, obtaining the equivalent ALD: $-[a\Delta b]-, a[c]-$.

The following remarks are rigorously justified for bounded ALD [17], but we expect them to hold for ALD with modulation too, with little change.

Ambiguity: The phenomenon of ambiguity occurs in ALD much as in CF grammars. For example the following ALD A ambiguously defines the Dyck language over the alphabet $\{b, e\}$: $-[\Delta\Delta]-, -[b\Delta e]-, -[\varepsilon]-$, because a sentence like *bebe* has two distinct trees in $L(A)$.

Other formal properties [17]: Bounded ALD languages (both tree and string) form a strict subfamily of CF. Actually an algorithm permits to construct a CF grammar structurally equivalent to a bounded ALD. More precisely the ALD tree languages enjoy the *Non-Counting* property of CF languages [18]. This property is believed to be a linguistic universal of all natural and artificial languages intended for human communication.

Considering the family of ALD string languages, we mention that it is not closed with respect to the basic operations of concatenation, star, union, and complementation: a lack of nice mathematical properties not affecting the potential uses of ALD for real languages, as witnessed by the successful definition of Pascal [20]. Indeed it is a common misconception that any language family is useless unless it is closed with respect to union and concatenation. Practical examples abound to the contrary, since very rarely it is possible to unite two languages without modification.

ALD models differ from Chomsky's grammars in another aspect: they are not generative models, because the notion of deriving a sentence by successive application of rules is not present. On the other hand an ALD can be used for checking the validity of a string by a parsing process. Preliminary analysis of the problem indicates that the classical parsing methods for CF grammars can be adapted to our case.

Grammar by example. An appealing feature is ALD suitability for *grammar inference*, the learning process permitting to identify a language in the limit [8] from a series of positive and negative examples. We already observed that it is straightforward to construct a LTD by inspecting a given sample of strings and extracting the k-grams. If the value of k is unknown, the learner can start with $k = 2$, then gradually increase the length if the inferred LTD is too general. Over generalisation means that some strings are accepted, which are tagged or negative by the informant. In the basic model of language learnability theory, sentences presented to the learner are flat strings of words, but other studies have considered another model, in which sentences are presented in the form of structures (such as stencil trees [16] or functor-argument structures in [11]). Availability of structure in the information has been defended on several grounds such as presence of semantic tagging. In practice, learning from structures is considerably simpler than learning from strings, because the problem space is much constrained. Constructing the ALD from a sample of stencil trees is a simple, determinate task that consists in extracting and collecting the relevant sets of k-grams, much as in the LTD case, since for a given integer k the ALD which is compatible with a given sample of trees is essentially unique. In the case of programming languages such as Pascal or C, small values of k have proved sufficient, so that this grammar inference approach can be applied without combinatorial explosion. As a consequence, ALD can be specified by examples, rather than by rules, because of the direct bijective relation

between two sets of positive and negative stencil trees and the ALD. In synthesis the k-gram extraction method is a good procedure for extrapolating an unbounded set of valid structures from a given sample. In contrast, for CF grammars there may exist many structurally equivalent grammars which are compatible with a sample of stencil trees, and the inference process is more complex and undetermined.

3. Mapping ALD on brain structures

From the point of view of neuronal modelling, associative language description has the great advantage of naturally blending into some of the most accredited theories of brain function. This is not the place to give a full account of the experimental evidence [4] on which these theories rest, for which reason we will only sketch some of the main results. We start by recalling the basic mechanisms assumed.

Cell assemblies (CA) [9]: It seems that the things and events of our experience have their representatives in ensembles of neurons (= nerve cells) which are strongly connected to each other and therefore (a) tend to become active all together ('ignite') even if only some of them are activated, and (b) stay active even after the external excitation ceases.

Sequences of cell assemblies: Although the individual CA , when activated, may follow a certain temporal order in the activation of its component neurons, this is not sufficient as a physiological basis of the sequential order which characterizes much of behaviour: sequences of items in some animal (bird song) and human (singing, speech) vocalisations, skilled behaviour such as occurs in crafts or musical performance. There must be control mechanisms in the brain which extinguish an active CA and ignite the following one in well defined sequences, although possibly with varying rhythm, which may be genetically determined or acquired by learning.

Synfire chains [1] [2]: There is evidence for very precisely timed sequences of neural activations in chains of neurons, or group of neurons, which conduct neural activity without the possibility of arresting it, or storing it in any place along the way. These so-called synfire chains are probably responsible for the timing of events within a time span of a few tenths of a second (whereas sequences of cell assemblies may have a duration of several seconds). These, too, are the result of learning processes.

We may ask some questions in connection with associative grammar. First, in what way are the sequences learned and stored which define the local rules of grammar (k-grams). The items which occur in ordered groups of k elements are words (though in the previous definitions they are denoted by single letters), composed of one or a few syllables, and therefore having a duration of between 0.2 and 1 (or at most 2) seconds (the duration of a syllable being about 0.2 seconds). A trigram composed of such elements would span a time of several seconds, too long for synfire chains but quite in the order of magnitude of various kinds of skilled behaviour (as in sports, musical performance, etc.). As in these other performances, the ability of the brain to learn sequences of events is evident and can even be related to some detailed neurophysiology in some cases. It is not impossible to imagine neuronal networks containing representatives of such learned sequences which are only activated when the correct sequence is presented in the input. They are stored in parallel and may be prevented from being activated more than one at a time by some mechanism of reciprocal inhibition. In analysing a sequence of

input events, they may also be activated in partially overlapping temporal episodes as required by the 'sliding' window idea. Of course the number of 3-grams is enormous in language, but the number of neurons involved in language processing in the brain is also very large, perhaps of the order of 10^8 and the number of the useful combinations of these neurons is possibly greater. Moreover, in reality we do not imagine the system to require that all k-grams have the same value k (as in the formal definition of ALD), but we rather suggest that the value will vary in different contexts, in order to minimize the memory requirements. It is conceivable that the minimal values required for k are discovered in the learning phase, assuming the grammar by example algorithms are deployed that we have sketched in Sect. 2. This mechanism for matching k-grams is absolutely essential for ALD as it is needed not only for recognizing constituents, but also for detecting the passage k-grams announcing the modulation between two constituents.

We notice that some k-grams play a special role when they act as initial (or final) k-gram of a constituent. The requirement of specially marking some k-grams is consistent with the so called 'X bar' theory of syntax [10], that affirms that each syntax class requires the presence of a specific lexeme. It is not difficult to imagine how the proposed neuronal scheme would detect such marked or compulsory k-grams.

Another question is how sequences of items are embodied in synaptic networks and how they are learned. Connections between neurons or groups of neurons are statistically symmetrical, but again in the cortical 'wiring', which is mostly stochastic, there is ample opportunity for asymmetrical connection that may embody the relation of temporal order or sequence. Moreover it is clear that synaptic relations between neurons in the cortex are to a large extent determined by learning (or experience), and it is certain that much of what is learnt in the way of succession of events (causal relations etc.) will determine unidirectional influence of a group of neurons into another one. There is a technical difficulty, however, in the physiology of learning when the sequence which is learned involves considerable delays, such as in trigrams (sequences of three words) which span over several seconds. We are forced to assume different delays interposed between input and internal representatives of trigrams. The observation of synfire chains already mentioned, may provide an adequate mechanism for delays with the possibility of representing asynchronous input in a synchronous way to the internal representative.

Finally the nesting rules (e.g. in Dyck's language of parentheses) have always been a problem when translated in neurological terms. A model based on the concepts of decaying activity in CA serving as quasi-bistable memory elements, has been shown to incorporate the virtues required for 'push-down' memories. The problem of insertion of phrases into phrases, to a certain extent involves the recognition of legal or unusual k-grams on the borders; this requires a brain mechanism which is able to interrupt the embedding phrase, for the time of the duration of the embedded one, followed by the resumption of the embedding phrase, after the completion of the embedded one. For this there are plausible neurological models, if the elements of the phrase are represented in the brain by concatenated CAs. Due to the nature of the CA, which is stable both in its active and in its inactive state, it is possible to imagine that the information for continuing the embedding phrase is preserved in the activity of the last active cell assembly before the interruption. In the case of several embedded constituents, there is the well-known fact of the inverse order of the closure with respect to the opening of the various

phrases. This is fairly easily explained by assuming that the activity of the CA which indicates an interrupted phrase slowly decays in time, and by assuming a rule by which the completion starts beginning with the most active CA, which will be the most recently activated one [15] [3].

Sequences of cell assemblies at a rate of $4-5$ a second have been postulated in many contexts of behaviour (e.g. vision) and are probably related to the periodic action of a mechanism controlling the state of activity of the cortex. This mechanism involves also inhibitory links which not only isolate individual cell assemblies from the background activity (by the principle of 'winner takes all') but also see to it that in sequencing cell assemblies one is just extinguishing when the next one is activated. This explanation essentially answers the question of how the 'place holders' of the theory are represented in the brain. But details are of course not known.

It is certain that not all elements in terms of which the grammaticality of a sentence is spelled out are identifiable with the elements at the surface level of language. Traditional grammar rules involve such things as grammatical (or lexical) categories. In theory we may think of tags that are attached to words by means of the all pervading associative mechanism typical of the cortex. But it is not at all clear how these categories are extracted from language in the early phase of learning.

A final note on the non-counting property that all human languages seem to have. Modulo-counting is on the other hand important in various forms of behaviour particularly in music: complex periodic structures occur in many compositions. The brain is quite effective in processing such periodic vocalisations or motor sequences as in percussion playing. We are therefore inclined to think that the non-counting property of language has to do with other constraints external to the brain, such as the excessive noise-sensitivity of such languages: a cough could easily transform an odd string in an even one, thus subverting the grammaticality and meaning of an utterance.

4. Conclusion

We believe that the associative language description model explains fundamental syntactic phenomena more convincingly, in terms of brain behaviour, than earlier attempts performed with context-free grammars. In spite of obvious limitations and drastic simplification, the model should be a good basis for extension and refinement. We summarise our findings and add a few remarks. The ALD model is based on constituent structures and on associative memory. If the first concept provides the basis of the classical Chomskian grammars, the latter has been rejected by linguists as inadequate for discriminating sentences from non-sentences. Yet associative processing is a fundamental mechanism of the brain and plays an important role in speech understanding and language production. The new model is based on the mathematical theory of local testability and counter-free languages, that has been investigated starting from the 60's, but had not been previously combined with constituent structures. Loosely related ideas have been studied in the area of programming languages, where the concept of precedence of operators was introduced by Floyd [7], to make parsing deterministic. He proposes that digrams (called precedence relations) be used to detect the left and right border of a constituent. The fact that non-counting operator precedence grammars can be easily

inferred from examples was noticed in [16] and the formal model was studied by [19].

Several mathematical aspects could be investigated in the vein of formal language theory. One such question concerns weak generative capacity: can any regular language be generated by an ALD? (the same question for CF languages had a negative answer in [17]). But a more relevant research project would be to assess the linguistic adequacy of the model, preferably by implementing the learning procedures that would allow ALD to be automatically produced from a linguistic body. Such a research program would permit to tune the local testability model, for instance to introduce the specification of compulsory k-grams, to allow Boolean operations on sets, or to specify long distance relations, etc. Concerning the brain model, the possibility to validate it we believe is based on timing analysis and comparison with psycholinguistic findings. Computer simulation of the proposed cell assemblies should also be feasible.

Acknowledgment. We would like to express gratitude to Alessandra Cherubini, Pierluigi San Pietro, and Friedemann Pulvermüller.

References

1. M. Abeles, *Local cortical circuits. An electrophysiological study*, Springer-Verlag, New York, 1982.
2. ———, *Corticonics. Neural circuits of the cerebral cortex*, Cambridge University Press, Cambridge, 1991.
3. V. Braitenberg, *Il gusto della lingua. Meccanismi cerebrali e strutture grammaticali*, Alpha Beta, Merano, 1996.
4. V. Braitenberg and A. Schüz, *Anatomy of the cortex. Statistics and geometry*, Springer-Verlag, New York, 1991.
5. N. Chomsky, *Syntactic structures*, Mouton, 1957.
6. N. Chomsky and G. Miller, *Finitary models of language users*, Handbook of mathematical psychology (R. Bush R. Luce and E. Galanter, eds.), Wiley, 1963, pp. 112–136.
7. R.W. Floyd, *Syntactic analysis and operator precedence*, Journ. ACM **10** (1963), 316–333.
8. E. M. Gold, *Language identification in the limit*, Information and Control **10** (1967), 447–474.
9. D. O. Hebb, *The organization of behaviour. A neuropsychological theory*, Wiley, New York, 1949.
10. R. Jackendoff, *X syntax: a study of phrase structure*, MIT Press, Cambridge, Mass., 1977.
11. Makoto Kanazawa, *Learnable classes of categorial grammars*, CSLI Publications, Stanford, 1998.
12. K. S. Lashley, *The problem of serial order in behavior*, Cerebral mechanisms in behavior (L. A. Jeffress, ed.), Wiley, 1951, pp. 112–136.
13. R. McNaughton and S. Papert, *Counter-free automata*, MIT Press, 1971.
14. J. E. Pin, *Varietes de langages formels*, Masson, Paris, 1984.
15. F. Pulvermüller, *Syntax und hirnmechanismen. Perspektive einer multidisziplinren Sprachwissenschaft*, Kognitionwissenschaft **4** (1994), 17–31.
16. S. Crespi Reghizzi, *Reduction of enumeration in grammar acquisition*, Second Internat. Conf. Artificial Intelligence (London), 1971, pp. 546–552.
17. A. Cherubini S. Crespi Reghizzi and P.L. San Pietro, *Languages based on structural local testability*, Combinatorics, computation and logic (C. S. Calude and M. J. Dinnen, eds.), Australian Computer Sc. Comm., Springer Verlag, 1999, pp. 159–174.
18. S. Crespi Reghizzi, G. Guida and D. Mandrioli, *Non-counting context-free languages*, Journ. ACM **25** (1978), 571–580.
19. ———, *Operators precedence grammars and the non-counting property*, SIAM Journ. Computing **10** (1981), 174–191.
20. S. Crespi Reghizzi, M. Pradella and P.L. San Pietro, *Conciseness of associative language descriptions*, Descriptional complexity of automata, grammars and related structures (J. Dassow and D. Wotschke, eds.), Universität Magdeburg, 1999, pp. 99–108.

21. A. Salomaa, *Formal languages*, Academic Press, 1973.
22. A. Wickelgren, *Context-sensitive coding, associative memory, and serial order in (speech) behavior*, Psychological Rev. **76** (1969).

Stefano Crespi Reghizzi, Dipt. Elettronica e Informazione, Politecnico di Milano, Milano, I-20133, Italy
E-mail address: crespi@elet.polimi.it

Valentino Braitenberg, Laboratorio di Scienze Cognitive, Universita' di Trento, Rovereto, I-38068, Italy *E-mail address*, lsc@inf.unitn.it:,
and Max-Planck Institut für biologische Kybernetik, Tübingen