Natural Language Parsing with Cell Assemblies: A Model of Nonconscious Human Language Processing

Christian R. Huyck
Middlesex University
c.huyck@mdx.ac.uk

Abstract

Cell Assemblies (CAs) are neural circuits that may be used as computational primitives. These primitives may be used to process natural language and particularly to parse language. CAs are based on knowledge of biological neural systems and have been proposed as the neural correlate of concepts. They are activated from direct sensory evidence or from other CAs and they compete with each other to become active. They are learned via simple local neural learning rules. Natural language parsing with CAs is done using a stack mechanism. It takes advantage of variable binding and semantic nets to execute this task. The binding and semantic nets emerge from the CA architecture. A CA-based model is used to explain parsing and other natural language processing tasks. CAs themselves are implicit representations of concepts. They are learned in an unsupervised manner. They are not necessarily accessible by the systems of consciousness; they are a Nonconscious and implicit phenomena.

1 Introduction and Background

Cell Assemblies (CAs) are reverberating circuits of neurons. They are the way humans (and other mammals) represent concepts and they can be used as the basis of an information processing model.

In biological systems, CAs are implemented in neural wetware. To understand CAs, we can inspect the wetware and we can develop computational simulations of neurons. In these simulations CAs emerge from the functioning of large numbers of neurons. Simulations and neuro-physiological evidence have shown how CAs can develop concepts. These concept/CAs can then be used to form the basis of more sophisticated processing.

This paper discusses CAs and how they may be used to parse natural language. CAs are implicitly learned and are a Nonconscious phenomena. Thus, this paper proposes a model of Nonconscious human language processing.

The next section of this paper is on CAs themselves, on how they are activated, how they compete with each other and on how they are learned. The third section is on parsing with CAs. It starts with word recognition, then explains grammar rules and finally how the parsing mechanism is implemented. The fourth section is on using CAs for full language processing. This is broken into subsections for semantics, one for other language modules, and one for combining modules. Finally, there is a section for CAs as an implicit representation and a section for CAs as Nonconscious phenomena.

2 Cell Assemblies

D. O. Hebb proposed a neural circuit model (16), a reverberating circuit of neural cells. Hebb called this reverberating circuit a *Cell Assembly* (CA). Some physiological evidence for CAs exists (e.g. (1; 9; 12; 35; 39; 40)).

While we have a good understanding of how neurons work, we have a comparatively poor understanding of how neural circuits work. Direct analysis of neural circuits is difficult because functioning circuits (in mammals) are very complex and our scanning techniques, including fMRI, PET and electrodes are not able to inspect the firing patterns of large numbers of neurons.

The basic CA model is a network of neurons; each neuron connects to other neurons, which in turn may connect back (directly or indirectly) to the original neuron. Some neurons (e.g. rods in the eyes) are not parts of circuits; however, physiological studies show that most neurons are in some sort of circuit (37). That is, the human brain is made mostly of circuits of neurons. Studying these circuits is crucial to our understanding of thoughts in general and language processing in particular.

There is a considerable body of theoretical and physical evidence for CAs. CAs have the following (theoretical) properties:

1. CAs recognise concepts.

2. CAs are composed of neurons.

3. Neurons may exist in more than one CA.

4. A concept is in working memory if and only if its CA is active.

5. A CA is a long-term memory item, and is formed by change in synaptic strength.

6. A CA remains in working memory for a short period (near 10^1 sec.).

7. CAs interact with other CAs.

2.1 CA Activation

A CA is activated when sufficient evidence is presented to activate it. This evidence comes in the form of neural stimulation. The simulation may come directly from the environment, in the form of sensory input, or from neurons in other active CAs.

Initially, some neurons in the CA are activated from either sensory evidence or from other CAs. The *dog* CA can be activated by seeing a dog or by thinking of a dog.

If enough evidence is present, then the neurons that are externally activated will start a cascade of activation. They will activate other neurons in the CA, which in turn will activate other neurons. This phenomena is called CA ignition. If a CA ignites, it will be able to sustain itself through reverberatory activation. The CA can remain active even if there is no external stimulation; a person can continue to think of the dog after it goes out of sight.

Neurons in the CA are equivalent to micro-features of an element of the category. Some features are more important than others; the neurons associated with these features have more inter-CA connection strength than those that are less important.

For CA ignition to occur, some evidence for that CA is needed. This evidence may come from a few important features being present, or a lot of minor features being present, or some combination of the two. Of course, if a lot of important and minor features are present than the CA will be easily activated.

No particular neurons need to be externally activated; that is there are no necessary conditions. This enables categories without necessary conditions as Wittgenstein (43) has described.

This also enables a CA to recognise an item it has never seen before. For instance, it may have never seen a particular object *C*. However, *C* has many properties that cups have, and the CA knows about these properties. Consequently, it can categorise *C* as a cup. This means that on seeing *C* the cup CA is activated.

2.2 CA Competition

Competition between CAs is essential to models based on large numbers of CAs. It is essential that only a small number of CAs are active at a given time. If a large number of CAs were active, they in turn would activate many more CAs, and the whole network would become active. Competition between CAs enables the network to focus on a small number of concepts.

Any given sensory environment provides evidence for a large if not infinite number of items. For example, when you are resting and looking at clouds or at abstract art, you can generate a wide range of interpretations. In CA terms, the cloud provides enough stimulus to activate the *elephant* CA in the correct environment. However it provides a great deal more support for the *cloud* CA. The *cloud* and *elephant* CAs are in competition. Since the *cloud* has more support, it typically is activated[2].

This type of competition is essential to survival. We need to select one interpretation quickly and it is important that that interpretation is usually correct.

Competition between concepts can be seen in many parts of the literature. For example, the Necker cube and lexical disambiguation (41) are examples of competition. Lexical disambiguation is a question of what actual meaning to choose when a homonym is presented. There is evidence for each of the homonym's meanings but one is finally chosen. The CAs compete until one wins.

Competition in CAs works via inhibitory neurons. A CA consists of both excitatory and inhibitory neurons. Both types of neurons are connected to other neurons in the CA, and neurons outside the CA. This is consistent with our understanding of human neural anatomy. The human cortex consists of excitatory and inhibitory neurons (2).

In a CA, the excitatory inter-CA connections are strong, and the intra-CA connections are weak. The inhibitory inter-CA connections are weak and intra-CA inhibition is strong. This enables a CA to suppress other CAs.

When two CAs have evidence for them they are competing to ignite. For example, on seeing the ambiguous word *lead*, all senses of the word are activated. That is all of the *lead* CAs get evidence and start the reverberatory cascade. However, as each CA gains strength, the inhibitory neurons within the CA also become active. These spread inhibition to the neighbouring CAs, and thus suppress the others. The competition ensues and one CA wins by inhibiting the other CAs. A working simulation of this competition within CAs can be found in (19).

2.3 CA Learning

One of the key points of CAs is that they are learned. This learning is accomplished by a change in synaptic weights known as Long-Term Potentiation (LTP) and Long-Term Depression (LTD). This type of learning can be seen in neurons, and has often been linked with the Hebbian learning rule (5).

[1] Fuster's experiments (12) implies that CAs last as long as 40 seconds. The exact time course of CAs is still in question. Time also varies from CA to CA; CAs that are closer to the environmental interface will function quicker. Of course CAs may also be turned off by other CAs. The importance of this property is that CAs remain active for several seconds; that is longer than a single neuron can be active, but is not long enough to represent a long-term memory.

[2] The *elephant* CA is only activated after the *cloud* CA becomes fatigued.

The standard Hebbian learning rule states that if neuron n_i contributes to the firing of neuron n_j then the connection $x_{i,j}$ tends to be strengthened. The anti-hebbian learning rule states that if one neuron fires and another connected neuron does not, then the connection strength is reduced. The Hebbian rule is linked with LTP and the anti-hebbian rule with LTD.

This change in connection strength is the only type of learning that is assumed in this paper. It happens in an unsupervised and local manner. All learning occurs as a response to firing neurons and neurons fire in response to the environment or in response to the firing of other neurons.

That explains the basic foundations of CAs. A CA is ignited when a subset of its neurons is fired by sources external to the CA. CAs are in competition with other CAs to ignite. CAs are learned in response to external stimulation and this learning is done by the change in connection strength of neurons.

As a primitive computational element, CAs are not necessarily consciously accessible. However, they are complex elements and can be used to explain language processing. As a first step, it is shown how CAs can be used to parse natural language.

3 Parsing with CAs

Some work has been done on how neurons in the brain might impose order on sentences (29). This section gives a broadened account of how this processing might be done.

For humans to parse sentences, they must recognise the words of the sentence, combine the words using grammar rules, and apply those grammar rules in a sensible fashion (an algorithm). CAs can be used to do all of these things, and the mechanisms explained below is consistent with existing psycholinguistic evidence (18).

3.1 CAs to Recognise Words

The first process in the chain is word recognition. Each word has a CA associated with it. The concept is related to its phonological representation; this word-concept pair is known as a bipolar pair (21).

The word is presented to the system. Neurons are activated via environmental stimulus (see section 2.1). Intermediate level CAs may be activated (phonemes or letters depending on the mode), but eventually the CA will become active. Evidence for the location in the brain of certain word CAs is provided by (28).

Recognition of a word is done by activating the phonological pole of the word's bipolar pair. This is error tolerant because all features of a word do not need to be present. Thus the misspelled *retreive* is easily recognised as *retrieve*.

Ambiguous words have a CA for each word-sense. If the input mode is visual, *lead* is ambiguous; if the input mode is auditory, *heard* and *herd* are ambiguous.

In the ambiguous case, both words are activated. The CAs suppress each other via inhibitory connections. Thus the ambiguous CAs are in competition and one will win the competition. Prior evidence, such as context or words earlier in the sentence, may help disambiguation (see section 4.2).

Word recognition is simple compared to grammar rule application. Computational models of CAs have recognised simple patterns (17). It can be easily imagined that a CA system could be developed that would recognise words. This system could easily be trained on instances of words where each instance consisted of either letters, phonemes or morphemes. It is likely that the human word recogniser takes phonemes and or morphemes as input, but also takes advantage of adjacent words.

3.2 CAs and Grammar Rules

Grammar rules are more complex than word recognition. Word recognition is simply categorising an instance of a word as a member of a class; this instance of *dog* is a member of the class of words *dog*. Grammar rules require recognition, but they also require application. Application requires binding elements to the rule, adding information, subtracting information, and binding the elements together.

Once one or more words are read, a grammar rule may be applied. For example, when the reader has seen the words:

The dog

Example 1.

the reader may apply a grammar rule like the traditional *Noun-Phrase* $\rightarrow$ *determiner and common-noun* rule. The grammar rule is implemented by a CA. The grammar rule must first be selected; that is the CA for the grammar rule must be activated. Once this is done, the rule must actually do something to change the state of the system.

Of course there must be some way to learn these grammar rules. Natural languages have different grammars and people have to be able to learn them.

3.2.1 Grammar Rule Activation

For a grammar rule CA to be activated, evidence for the presence of the grammar rule must exist. In example 1, the evidence does exist as the CA for *The*, a determiner, exists; and the CA for *dog*, a common noun, exists. That is, the grammar rule CA is activated largely by the word CAs.

This is an instance of a CA that encodes both hierarchy and sequence. In section 3.3 it is shown how the grammar rule is also involved in processing. The CA encodes hierarchy because it is binding a lexical class of words. It is encoding sequence because it needs those lexemes in a certain sequence. When a rule is applied, it is active and its constituents are active. The rule application binds them all together.

As with ambiguous words, multiple grammar CAs may be activated. This may be due to local or global ambiguity in parsing. As with ambiguous words, these grammar rules will be in competition. Thus one will be selected based on the evidence that is present.

Evidence may be more than mere syntactic evidence. When words are activated their lexical category is activated, but also their semantic content is activated. When the string

saw the girl with the telescope
Example 2.

is read, there is an ambiguity between attaching *the telescope* to *the girl* or directly to *saw*. Semantics can be used to disambiguate this by choosing the appropriate grammar rule.

Rules can be activated to match traditionally ungrammatical phenomena. The ungrammatical sentence *I be good.* is easily understood. While the person constraint between *I* and *be* is not met, enough evidence is present to apply an appropriate rule. This is similar to the case of misspelled words.

3.2.2 Grammar Rule Application

When people parse sentences, the application of grammar rules is more than a simple syntactic check. Grammar rule application builds a structure that is used in later language processing and may remain around when the sentence, paragraph or entire text is completed.

CAs have an explanation for working memory (CA activation) and long term memory (synaptic change to make a CA). However, parsing requires an other type of memory, variable binding.

It is reasonable to assume that people have CAs for individual words, but it is unreasonable to think that people have CAs for all adjective, noun combinations that they will ever encounter. Consequently, there must be some way to bind these CAs together. This variable binding mechanism could be used to combine existing word CAs while parsing.

One proposed mechanism for variable binding is pattern oscillations (25). CAs that are bound together fire in similar patterns. This may be supported by working memory buffers; CAs that bind other CAs together making them oscillate in the same pattern.

When two CAs are bound together any connections between them will be strengthened. Since neurons in both CAs are firing at similar times, the Hebbian learning rule will strengthen connections between the neurons [3].

Applying a grammar rule binds constituents together and adds other information. Binding is simply a matter of making all the neurons fire at roughly the same frequency. A group of neurons can easily regulate firing (25). Leaky integrator neurons in a binding site act as frequency modulators and thus change the frequency of the firing of neurons in the bound CAs. This brings them into the same rhythm.

In the parsing model described below, the binding site is a stack element. The stack element binds the constituents and the rule together. Activation of the rule CA activates the stack element. The rule ceases to be active after it is applied, but the stack element remains active along with important elements of the bound CAs.

In addition to activating the binding stack element, the grammar rule may also add information. For example, the grammar rule $S \rightarrow NP\ active\text{-}VP$ will provide the information that the *NP* is the actor.

Applying the grammar rule to *NP-I* and *active-VP-lead* (from example 3 below) leaves several pieces of information bound together. That is, several CAs are active, they are firing at the same frequency, are in a self-activating loop with the binding site and are strengthening their internal connections. These CAs include *I*, actor, *lead* and action.

Important words remain active for a long time, and this supports synaptic change via a Hebbian learning mechanism. This may be sufficient to push from working memory via variable binding to long-term memories, or an extra mechanism may be needed.

This is a sketch of one possible implementation of rule application in CAs, but the understanding of rule application in CAs is in its infancy. I know of no working simulations of rule application in CAs.

3.2.3 Learning Grammar Rules

Grammar rules involve both semantic and syntactic information. How are they learned? They are learned by a combination of both semantic (26) and syntactic (13) bootstrapping. Simple rules such as sentence from actor-noun action-verb are initially learned by repeated combination of experiencing actors and actions; for example *dog eat*.

Once this simple rule is learned, it becomes easier to learn new words. *ran* is an action because *the dog ran*. Knowing this simple rule supports the learning of more sophisticated syntax.

Lexical classes are developed as more abstract CAs. For example, these are things that can do something, that is nouns.

In some sense, this has introduced hierarchy as another mechanism that CAs can implement. That is, CAs can be in some sort of hierarchy. CAs should hold some sort of family relationship with each other. This type of category (34) seems to be humans' categories are related. The CA mechanism should be able to learn these relationships with no new mechanism aside from Hebbian learning. I am unaware of any CA-based simulation that can

[3] CAs not bound together but firing will be less likely to be strengthened because they are not firing at the same time due to frequency differences. Of course, they will fire simultaneously occasionally, so the connections will be strengthened but much less so than bound CAs.

currently generate this type of family resemblance hierarchy, but am in the process of designing an experiment to implement hierarchies with CAs.

Up to now, this paper has only assumed that the brain is initially an unorganised mass of neurons. Clearly the brain has some structure to it and this structure is genetically programmed. While the brain may have a relatively complex macro-structure, the micro-structure is not well specified (2). Genes determine that the brains has two hemispheres, but genes do not determine the connection pattern of neurons. These connections seem to be random based on a few parameters such as distance biasing.

This digression into brain morphology is brought on by language learning and Universal Grammar principles (4). There is wide agreement that there is some inbred capability for language processing in humans, a language instinct (27). It is not widely agreed on what exactly is inbred, but whatever it is must manifest itself as some phenomenon in the brain. It seems reasonable to assume that genes can determine brain macro-structure. Consequently, it seems reasonable to assume that the brain has special areas for language processing and there is a great deal of imaging evidence for this (e.g. (28)). CAs can grow in these areas to set language specific rules and concepts.

Genes build the basic language modules and these modules have connections. CAs grow in and across these modules to set language specific parameters. CAs also cross into non-language areas to impart more semantics to language phenomena. Language is learned within some largely predetermined macro-structure (i.e. Universal Grammar); however the micro-structure is largely undetermined and is learned via CA formation.

3.3 CAs and Parsing

In the proposed model, parsing proceeds via a stack-based mechanism. The stack has a small number of elements. Elements appear on the stack (initially via word recognition). Grammar rules are applied to the elements and combine stack elements reducing the size of the stack and adding information.

Variable binding is used by grammar rules to combine CAs. Variable binding uses special CAs for support. These CAs are the stack elements.

The stack elements themselves are neural hardware that set oscillations patterns of existing active CAs. When a grammar rule is applied, the stack element associated with the first constituent of the grammar rule keeps its pattern. It also sets the pattern of the other stack elements used by the rule to the pattern of the first stack element.

For example, element 1 is binding CAs together, and is oscillating at pattern 1. Elements 2 and 3 are oscillating at patterns 2 and 3. A grammar rule is applied combining elements 2 and 3. This new element 2 oscillates at pattern 2, but now is bound to critical constituents of element 3. The critical constituents of element 3 are now oscillating at pattern 2. Stack element 3 is now free for the next word.

This type of stack-based parsing system works in existing Text Engineering applications (18). With CAs, the stack element is responsible for binding the essential parts of its constituents. Clearly, some words will be lost in large structures. They will no longer be bound, and their CAs will become inactive. The crucial CAs will remain active, and these may even enter long-term memory via LTP and thus via CA formation.

The location in the brain of the stack elements that bind CAs is not known, however (38) points toward the hippocampal system. There may be general binding areas in the hippocampal system and specialised language binding areas in the language areas of the brain.

3.3.1 Parsing Example

An example of parsing a full sentence is given below. This sentence has lexical and syntactic ambiguity.

I lead the orchestra with a baton.

Example 3.

Initially the stack is empty, but since the system is processing a sentence, it knows that it is looking for a sentence. *S/S* is put on the stack[4]. This stack element is oscillating at say 30Hz. Whenever information is added to a stack, that merely means that the CA for that information is activated, is in an activation loop with the stack element, and is oscillating at the same frequency. When information is removed, the CA becomes inactive, though traces may be left due to changes in synaptic strength brought on by activation.

The first word *I* is recognised. It is possible that a roman numeral is also activated but it is likely that *I* is recognised as a pronoun.

The pronoun is put on the stack. Stack mechanisms can be implemented in many ways with CAs but this example will use a very simple mechanism with one CA buffer for each stack element. This buffer has the function of making all the CAs attached to the buffer oscillate at the same frequency, thus binding them together. It also encourages them to stimulate each other, by keeping them more active. In this case, the pronoun *I* is put on the stack and oscillates at say 40Hz.

Once the pronoun is on the stack, the rule *NP* → *pronoun* will be activated. This rule will be applied and will bind properties to the second stack element and thus to *I*. These properties include complete-NP.

With *S/S* and *NP* on the stack, the *S/VP* → *S/S & NP* rule can be applied[5]. It may be the case that an *NP* is not a good head noun (e.g. yesterday). However, *I* is a good head noun, and it is a complete *NP*, so the rule

[4]The slash is a way of symbolising missing information and is often used for gapping phenomena. Here it is used to say there is a sentence *S*, but we still need a sentence

[5]The rules used in this example are right leaning (32). This mechanism can be used for general context free grammar parsing but right leaning rules keep the size of the stack small. A small stack size is crucial because it reduces the number of elements in working memory.

is applied. This removes the information that the stack element looking for a sentence, and adds the information that it is looking for a head *VP*. This also changes the oscillation pattern for *I* from 40Hz to 30Hz.

Since no other rule is active, the next word will be attended. *lead* is ambiguous and all senses of the word are activated (41). The CAs of the different senses are activated and are in competition. Since a head verb is being sought, the verb sense of the word receives extra stimulation and wins the competition. Now that the *active-transitive-V-lead* CA is active, it is put on the stack[6].

The *VP* → *active-transitive-V* rule is activated and applied. Since *lead* is a transitive verb, the information that it needs an *actor* and an *object* is added, and the information that extra complements can be accepted is added. This frame-based (11) approach enables argument structure to be easily integrated. Additionally, this frame is connected with the word *lead* so it has some information about the type of actors, objects and other complements that *lead* takes.

The stack now has *S/VP* and *VP*. This activates the rule *S* → *S/VP and VP*. This rule is applied and has the function of binding *I* as the actor of *lead*. Now element 1 is *actor-I action-lead S* oscillating at 30 Hz.

No other rule is sufficiently activated, so the next word will be attended. *the* is recognised and the rule *Common-NP/head-noun* is applied. This puts the *Common-NP/head-noun* on the stack oscillating at 40Hz.

No other rule is activated so the next word is attended and *N-orchestra* is added to the stack as the third element. This element is oscillating at say 50 Hz. The rule *NP* → *Common-NP/head-noun and N* is activated. At this point this rule may fire or may be delayed to assure that *orchestra* is the head noun. Assuming *orchestra* is a good head noun, the rule is applied, leaving *S-lead* and NP-orchestra on the stack.

Again, the rule *S* → *S and NP-object* can be applied. However, the NP may not be complete. It may have extra complements. In this case the system waits for extra information[7].

The next word *with* is attended, and is added as a preposition to the stack. Both the *S* → *S and NP-object* and *PP/NP* → *prep* are licensed by the stack. Since a *PP-with* can be either an argument to *lead* or modifier of *orchestra* the system is not sure that it is validated. It applies the *PP/NP* → *prep* rule, leaving three elements on the stack.

a is read, recognised as a determiner and put on the stack. The rule *PP/head-noun* → *PP/NP and det* is applied again leaving the stack with three elements.

The last word *baton* is added to the stack. The rule *PP* → *PP/head-noun and N* is applied and three elements are left on the stack.

Now two different rules are validated, the PP could modify *orchestra* or *lead*. Since *baton* is a good instrument for *leading* particularly when an *orchestra* is being lead, that rule wins. This means that the *S* → *S and NP-object* rule is applied to the first two elements of the stack. This has the effect of modifying the stack and binding information together. Changing the stack is done by combining element one and two and moving element three to element two[8].

The applications of this rule also binds *orchestra* to the object slot of *lead*. A weakness in this model is that *I*, *actor*, *object*, and *orchestra* are all oscillating in the same pattern. How is *I* bound to *actor*, and *orchestra* to *object*? They may be bound by semantic relationships and even by short-term memory changes during rule application. Still a frame-like structure seems necessary (see section 4.1).

The rule S → S and PP-instrument is then applied. This binds the *baton* as the instrument of *lead*. Finally, the period is read, and the sentence is completed.

With this particular stack mechanism and oscillation rates, the system can always tell what is the top of the stack and the order of the stack. In this case, the stack is ordered by frequency of oscillations. However, the oscillation patterns could be different. The important part is that the stack elements are always in the same order. There are a small number of stack elements so there is no need for an infinite amount of hardware. One of the problems with this is that some stack elements might fatigue. This may be one reason to have a more elaborate stack mechanism. Still this mechanism is a reasonable first approximation.

This mechanism is highly speculative and many parts of it can be changed. However, it is a CA-based, and thus a neural-based, mechanism. As such it is a model of neural language parsing that is consistent with existing linguistic, psycholinguistic and neural evidence.

3.3.2 Backtracking and Failure to Parse

A strong metric for a model is that it fails when the system it is modelling fails. The model described in this paper models human language parsing; human parsing, though incredibly sophisticated, frequently fails. In particular there are three types of failure that are quite important, misanalysis leading to backtracking, failure to analyse traditionally grammatical parsing from garden pathing, and failure to analyse traditionally grammatical sentences due to centre-embedding.

[6] Putting a word on top of the stack consists of finding the top of the stack and binding the word CA to that stack element. Finding the top of the stack can be done by looking through the stack elements until one is found that is not active. This could be done serially, but it is probably done in parallel. Also a stack top CA, could point to the top. Once the top is found, binding is done as described earlier.

[7] Another way to describe this is that the object is attached, but that certain sites remain active. This seems like a viable interpretation, but in this case, one can say that an extra stack is needed for the active sites. For simplicities sake, I will not apply the rule yet, but humans certainly have some idea that *orchestra* is the object at this state.

[8] Again, other mechanisms are possible, but this one is close to standard stacks.

Misanalysis in parsing happens frequently[9]. A person is reading through a sentence, comes to an ambiguity, chooses incorrectly, and later in the sentence recognises they were wrong. They usually then backtrack into the sentence, and choose the correct interpretation. This is shown by eye-tracking experiments (20). Example 4 is a sentence that might cause backtracking.

While Mary mended a sock fell on the floor.

Example 4.

Initially *sock* is attached as the object of *mended*, but when it is noticed that it is the actor of *fell*, the reader backtracks, repairs the problem and continues on.

This model would fail in a similar way if it made an incorrect parsing decision. Backtracking itself would require an extra mechanism (22). However, this mechanism could be implemented in a rule-based system and this rule-based system could be implemented in a CA-based system.

A garden path sentence is one where a decision is made when an ambiguity exists, but backtracking can not fix the problem. Example 5 is a syntactically valid sentence but people usually fail to assign it an interpretation.

The teacher taught by the Berlitz method failed.

Example 5.

The key here is that *taught* is something that a *teacher* usually does, so it should be the main verb of the sentence. However, teachers also can be taught. If the word *teacher* is replaced by *student*, people normally parse it correctly the first time.

This parsing model would fail on example 5 in the first pass. If there was not enough activation for the relative clause reading of *VP-taught* on the second pass, then this model would fail to read the sentence just as many people fail.

The third phenomenon is centre embedding. Example 6 is a centre embedded sentence.

The mouse the cat chased escaped.

Example 6.

Generally people can assign a meaning to this. It is centre embedded because the sentence *the cat chased the mouse* is embedded in the sentence *the mouse escaped*. A further embedding leads to problems.

The mouse the cat the dog smelled chased escaped.

Example 7.

Example 7 is a doubly embedded sentence. It is traditionally grammatical, but people find it very difficult to assign an interpretation. The reason seems to be that too many stack elements have been used. Traditional context free grammar descriptions of language do not account for these types of memory limits. Marcus (24) suggests that we have three stack elements. The model described in this paper could easily incorporate such limits[10].

This model can parse language. Additionally, it fails when humans fail. This is good because it now parses like humans. This is good for cognitive modelling, but it is also good for Text Engineering because a system that functions like humans is more likely to generate the same results as humans. In the case of language, the humans' result is the correct one.

4 CAs for Full Language Processing

A CA-based model of natural language parsing has been described. CAs could be used to describe other aspects of natural language processing (NLP). This section first shows how CAs can be used for semantics and to enable learning of events. It then shows how a CA-based model could improve other traditional NLP modules. Finally, it shows how a CA-based system could implement sophisticated communication between CAs and between processing modules.

4.1 Semantics and Learning

Clearly semantics is already built into this CA-based model. Each word that is activated has semantic content. It is a symbol grounded in the real world (15). These semantics can be used to make parsing decisions and they can be used to leave long-term semantic memories.

Parsing decisions need semantics (6; 18). In example 3, *with the baton* is bound to *lead* because it is a good instrument for leading. If, on the other hand, the sentence had been example 8,

I lead the orchestra with tuxedos.

Example 8.

with tuxedos would have been bound to *orchestra* because *tuxedos* are easily associated with *orchestras* and they are poor instruments for *leading*. This semantic decision influences CAs by connection strength. The CA for *NP-orchestra* and *PP-with-the-baton* is connected to the *NP* $\rightarrow$ *NP and PP-modifier* rule but does not send much activation so this rule is not activated. *NP-orchestra and PP-with-tuxedos* does send a lot of activation so the *NP* $\rightarrow$ *NP and PP-modifier* rule is activated.

The *orchestra-tuxedo* pair knows to send activation because the system has learned that *orchestras* often have *tuxedos*. It has done this by seeing instances of this pair or related things (e.g. *symphonies* with *tuxedos*). There is not an *orchestra-tuxedo* CA, but the CAs are connected in a semantic net[11].

One of the properties of human language processing is that humans can learn from reading a sentence. Consequently, a complete account of human language processing would explain how a reader could put the results of the semantic interpretation of a sentence into long-term memory; i.e. how would you learn a sentence.

[9] I hope it has not happened to you wile reading this paper, but I find myself frequently backtracking while reading the newspaper.

[10] A full model of these phenomena would require a more sophisticated stack mechanism.

[11] This is a new type of thing that CAs can do. CAs can implement semantic nets in addition to hierarchies and rules. See section 4.3 for arguments why CAs can do these things.

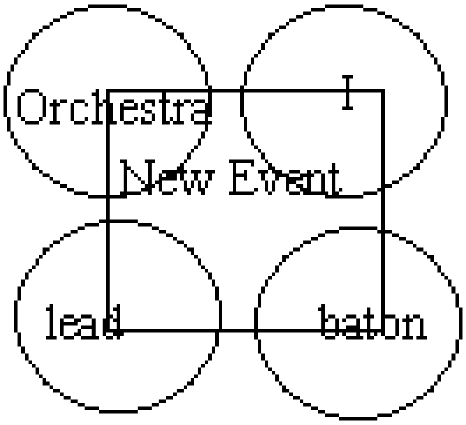

Figure 1: Neurons in New Event-CA

A model exists of how a transient pattern of rhythmic activity representing an event can be transformed rapidly into a persistent and robust memory trace as a result of the change in synaptic weights (38). This model is not based on CAs but is consistent with the CA model.

It is likely that a full-fledged model of NLP using CAs will take advantage of the participation of a neuron in multiple CAs to explain this rapid memorisation. Learning a new concept is difficult because a large number of synaptic weights need to be changed to learn that concept. Learning a new event is comparatively easier because the event consists of relationships between existing concepts/CAs.

CA theory states that a new concept needs to be represented by a new CA. This means that a new reverberatory circuit needs to be developed through Hebbian learning. Since the event is only described once (in a sentence), the new event-CA must be learned quite quickly. The event has elements in it that already have existing CAs. The new event CA can take advantage of these existing CAs.

In example 3, a new CA for the event would be needed if the system is going to remember the event. The neurons in the CA could consists of some neurons of the elements and some new neurons. The neurons from each individual constituent would be closely connected before the sentence is read. Building the new event-CA would be easier because sub-assemblies would already exist. Figure 1[12] shows the pre-existing CAs and the new CA learned in reading example 3. All that is needed is a minor change in the weights of connections in pre-existing CAs and changes in the new neurons.

Modification in the neurons of the pre-existing CAs may be needed to prevent the old CA from becoming active when the new event-CA is activated. Of course, modifications of those neurons is also needed to assure that these neurons are linked to the new neurons in the event-CA. However, these modifications should be minor in comparison to the modifications to learn a new concept CA from scratch.

One of the hallmarks of semantic theory is the case frame (11). An event has several slots and these slots may be filled explicitly or by default knowledge. In example 3, the verb *lead* has three slots, actor, object and instrument, and these slots are filled by *I*, *orchestra*, and *baton* respectively. How might this be done with CAs?

When a verb is processed, certain information is already known; this default knowledge may be omitted in the sentence. Also the slots are already known; e.g. *sleep* does not have an object. This knowledge is part of the verb's CA.

As new slots are filled, the fillers are bound to the slot part of the verb CA. While all the slots and fillers are oscillating in the same pattern, the actual neural connections between the filler CA and the appropriate part of the verb CA.

4.2 Other Language Processing Modules

Traditional NLP systems are broken into several modules (8). These systems can be used for many things but currently fall far short of human performance on complex tasks such as translation between languages (30). These systems have different modules but a typical system might have four modules: word recogniser, parser, co-reference resolver and discourse analyser.

All of these modules can be implemented by a CA-based system, and a CA-based system could improve on the performance of all of them. The primary means that a CA-based system could use to improve on these is by allowing them all to work together.

CAs are activated by evidence in the form of neurons that are part of the CA being activated. If enough of those neurons are activated (enough evidence is present), a self-stimulating circuit will be activated.

In a given module, most input would come from the traditional evidence. For example, evidence for activating a rule CA comes from words (or constituents) that are part of it. Evidence may also come from other active or recently active CAs. Thus a context mechanism is already built into the system. Similarly, lexical tagging can take advantage of prior context because it is already built into the system.

This type of inter-module communication is further discussed in next section (section 4.3). This communication is a major advantage of a CA-based model but other advantages exist for each of these modules.

It has already been shown (section 3.1) how words can be recognised. A CA-based system has the advantage of context, of error tolerance and of the ability to learn new words.

Most lexical lookup systems fail if a word is misspelled. The error tolerant nature of CAs allows for misspellings.

[12]This picture is meant to represent set relations of neurons. It has no implied correlation to actual brain topology.

A CA-based system could also introduce new words and new meanings for known words. A new word, if sufficiently important, would be created by creating a new CA. This would have all the evidence from the text to build that CA. New meaning for words could also be generated when existing meanings do not quite match. A process of CA fractionation could be used for this; that is, the old CA would break into two parts that would be two new CAs.

Parsing has already been described. Co-reference resolution could be implemented by searching active CAs. When a pronoun occurs it usually needs to refer to something. It usually refers to something that has already occurred and it has some constraints. For example, the pronoun *he* must refer to a male. A standard solution is to search through items that could fulfil the role and the nearest is chosen. This heuristic is quite successful, but does frequently fail. Attempts have been made to include importance in this decision because an important item is more likely to be the reference of anaphor. Importance is already built into CAs by activation and persistence. Something that has been active for a long time is important. Also anaphor may allow a CA to persist longer by spreading activation and allowing the CA to recover from fatigue. In turn, something that has already been referred to anaphorically is likely to be refereed to again.

Discourse analysis is probably the least understood system. Most functioning NLP systems are domain specific. Domain specific information from sentences are stored and combined to form larger discourse items. More general systems combine all of the information, add default knowledge and may even handle implicatures (14). Many early NLP system took advantage of scripts (36) to handle default knowledge. In any reasonable length text their are a great number of concepts and there is a combinatorial explosion of possible relations between them. This is a problem for traditional algorithms.

A CA-based system could use default knowledge and scripts in a fashion that is an extension of frames. However, unlike frames, scripts could take fuller advantage of CA formation.

Persistence of CAs is particularly important here. When reading a text of a paragraph or a few paragraphs, a large number of CAs will become active. CAs that are central to the text will be repeatedly activated. The longer a CA is active, the more likely that something will be learned [13].

A CA-based system can say something about which CAs are important in a given story. A discourse analyser based on CAs would thus remember the important things.

4.3 Modularity and Connectivity of a CA System

Already in this paper, CAs have been used to inhibit each other, to bind other CAs together, to implement rules, to implement hierarchies and to implement semantic nets. This is in addition to the basic function of CAs, to represent concepts. How can CAs do all these things?

The answer is, of course, neural connections. Yet how can a CA support all of these connections; a CA will have connections to a large number of other CAs via the semantic net alone[14]. A dog-CA would have *IS-A* connections to mammal, house pet and other categories; it would have its own subcategories of Collies, Retrievers Terriers and others; it would have *Consists-Of* connections to paws, tails and wet noses; it would have connections to cats as enemies dog food and leashes; it would have connections to instances of dogs a person knew (possibly via subcategory CAs); it would have links to the fear-CA if a person was afraid of dogs. Clearly there will be a large number of inter-CA connections.

A CA can support all of these connections because a CA has a huge number of neural connections. Assuming that a CA has 50,000 neurons [15] and that each neuron has 1000 connections (23), then each CA has 50,000,000 connections. Of course many or even most of these connections are to other neurons in the CA. Still this leaves 20 million neural connections per CA. The connection from dog to mammal may need many neural connections but this still leaves room for a large number of semantic net connections (and other types of connections) between CAs.

Neurons participating in multiple CAs will also increase the number of connections. If a neuron is active and it is in two different CAs then it will be evidence for both of them. Indeed there are so many connections that one might ask how a CA-based system can prevent all the neurons from becoming active.

It should be noted that connections are distance biased (37); neurons are likely to be connected to near by neurons but not to a given distance neuron. This distance biasing is a key component of CA theory because it enables enough activation to allow CAs to form. However, it also implies that CAs are localised.

Evidence exists that some CAs are not localised (28). These two points can be reconciled by noting that sub-CAs are localised but are combined via long-distance neural connections (5) and by long chains to form CAs that are distributed across large areas of the brain. A CA consists of many sub-CAs.

The brain seems to be broken up into a large number (around 50) modules. Many of the modules, possibly all, are quite domain specific. Blackboard systems are popular symbolic AI mechanisms that allow different AI modules to communicate (31). The CA model already has this communication built in. A CA-based system can easily integrate lexical modules, parsing modules, discourse processing modules and others.

[13] Neural fatigue may be a component in learning. When CAs are active for a long time their neurons fatigue.

[14] Category hierarchies are just part of the semantic net and as such will be treated as part of the semantic net in the rest of this discussion

[15] There is not much evidence on the size of a CA, but this number is a good starting place.

CAs may cross these module boundaries and connect the modules. Thus within a module, a CA can do its normal work. It also spreads information to other modules. If that knowledge is needed by another module, activation in the module may boost the activation in that portion of the CA. This will reinforce the CA in its base module and maintain its activation for longer. If the information in a module is not needed, activation in the CA in that module may continue but may fade away.

CAs support robust communication between modules. They, along with modular brain architecture, implement a blackboard-like system, where the blackboard is the CAs that are active.

The sceptic can easily note that this is really a lot of theorising. There is little neuro-psychological evidence and there is no working simulation for most of these complex structures. Clearly this is the case, and work needs to be done in these areas to further explore the problems. However, there is no reason why these things can not occur. Indeed, they form the best explanation I know of because they are based on solid evidence. A description of the brain as a symbol-processing mechanism is reasonable, but how do these symbols emerge. They must somehow be based in neurons.

Fodor (10) points out that humans do a great deal of abductive reasoning. That is reasoning is not always modular. CA theory forms a basis for this abduction because CAs cross many modules. Symbolic reasoning is still possible due to variable binding.

Modules can exist, but the boundaries are more porous than in the usual symbolic system[16]. Modules in the brain may be a response to input, or a response to macro problems. There are modules for vision. Other modules may have emerged from Darwinian pressures. If a language module was ever genetically stumbled upon, it would have conveyed a huge advantage to the people who had this structure.

Despite being over 50 years old, CA theory is in its infancy. There is a great deal we do not understand about CAs and how they interact. However, CA theory is a sound basis for a cognitive architecture. It can explain complex phenomena, it is based on sound neural principles, and it is testable.

5 Learning CAs as an Implicit Representation

CAs are an implicit representation of a concept. For instance, the grammar rule for a *NP* $\rightarrow$ *determiner and common-noun* is not represented in any explicit form, but is instead represented by a series of well-connected neurons.

CAs are learned implicitly. They are learned unconsciously (though the learner may or may not know he has learned something) and they can learn abstract knowledge such as grammar rules. These are the requirements for implicit learning (3).

One of the classic implicit learning tasks is sequence acquisition. This is a problem that was studied in many early CA experiments (e.g. (33)). CAs have a close relationship with implicit learning.

It is difficult to test for the presence of a CA even in a neural simulation. However a good CA detector could make the presence of CAs explicit given access to the neurons and their connections (42). A CA detector might measure connection strengths between neurons to determine the presence and location of a CA of CAs in a given network.

While CAs can be measured in the laboratory, this does not mean that humans can test their own CAs. CAs are not open to introspection.

In many symbolic parsing systems, rules exist to select between ambiguous grammar rules (18). This is an explicit representation. In the proposed CA-based parser, grammar rule selection is even more implicit than grammar rules. It is represented not via CAs; instead selection emerges from the architecture.

Moreover, learning a CA is usually an unsupervised and implicit process. A CA is merely a bunch of features that tend to hang around together. This is also what a concept is. Instances of the concept are presented, and this stimulates neurons (features); if neurons are co-active, the connections between them will be strengthened. If a set of neurons has been co-activated often enough, their connections will be strengthened and they will form a CA.

This is unsupervised but it can also happen implicitly. Attention certainly aids CA formation, but formation can occur without it.

6 CAs & Language Processing: a Nonconscious Phenomena

CAs exist in all portions of the brain. The systems of consciousness (7) do not have direct access to all CAs. This is why introspection is such a dangerous psychological tool.

Many linguists research grammar rules. If grammar rules were open to introspection, the correct grammar rules would have been known long ago. Instead, we are studying human language processing to discover how it works. We must use clever tools (e.g. eye-tracking experiments (20), and fMRI) to gain access to these systems.

Some CAs may be more open to consciousness than others. The CA for *dog* may be more consciously accessible than a CA for a grammar rule. Perhaps this is because the *dog* CA is richer having links to our ontology as well as all of the senses. A grammar rule CA is something we

[16] Modularity is an excellent tool for designing because it enables the designer to break the problem down into pieces that do not interfere with each other. Solving each piece individually is easier than solving the whole problem. However, the brain was not designed so it does not need modularity.

never even learn about explicitly until we are in adolescence. On the other hand, this may be due to a grammar rule CA being embedded deeply in the language processing system.

CAs are Nonconscious. This paper has shown how they might be used to process natural language. This gives a plausible explanation why so much language processing is Nonconscious.

CA theory largely exists as theory. There is evidence for CAs in the neuro-psychological literature; computational simulations have shown simple functionality but nothing approaching the complexity of language processing. Understanding how millions of neurons work together is a complex task and one that will not be completed in the near future.

This paper has shown how CAs might process language. Evidence that our neurons process language this way is scant. However this model is a starting point for a computational, neurological and psychological linguistics. Undoubtedly parts of this straw man are wrong, but at its heart, in the use of CAs, it is correct.

References

[1] Abeles, M., H. Bergman, E. Margalit, and E. Vaadia. (1993) Spatiotemporal Firing Patterns in the Frontal Cortex of Behaving Monkeys. *Journal of Neurophysiology* 70(4):1629-38

[2] Bratenberg, V. (1989). Some Arguments for a Theory of Cell Assemblies in the Cerebral Cortex In *Neural Connections, Mental Computation* Nadel, Cooper, Culicover and Harnish eds. MIT Press.

[3] Cleeremans, A. (1993) *Mechanisms of Implicit Learning: Connections Models of Sequence Processing*. MIT Press. 0262032058

[4] Chomsky, N. 1980 *Rules and Representations* New York: Columbia University Press

[5] Churchland, P.S. and T.J. Sejnowski (1992) The Computational Brain. MIT Press.

[6] Crain, S. and M. Steedman. (1985) On not being led up the garden path: the use of context by the psychological syntax processor. In Dowty, D., L. Kartunnen and A. Zwicky (eds.) *Natural Language Parsing: Psychological, Computational and Theoretical Perspectives.* New York: Cambridge University Press, pp. 320-358.

[7] Crick, Francis. (1995) *The Astonishing Hypothesis* Simon and Schuster.

[8] Cunningham, H., Y. Wilks, and R. Gaizauskas. 1996. GATE: A General Architecture for Text Engineering. *CoLing 1996*

[9] Eckhorn, R., R. Bauer, W. Jordan, M. Brosch, W. Kruse, M. Munk and H. Reitboeck. (1988) Coherent Oscillations: A Mechanism of Feature Linking in the Visual Cortex? *Biological Cybernetics* 60 pp. 121-30

[10] Fodor, J. (2000) The Mind Doesn't Work That Way: the Scope and Limits of Computational Psychology. MIT Press. 0-262-06212-7

[11] Filmore, C. (1968) The Case for Case. In *Universals in Linguistic Theory* E. Back and R. Harms (Eds.) Holt, Rinheart and Winston Inc.

[12] Fuster, J. (1995) An Empirical Approach to Neural Networks in the Human and Non-Human Primate.

[13] Gleitman, L. (1990). The structural sources of verb meaning. *Language Acquisition I*, 87(1):3-55.

[14] Grice, H. Logic and conversation. (1975) In Syntax and Semantics, Vol. 3 Speech Acts. p. Cole and J. Morgan (eds) New York: Academic Press

[15] Harnad, C. (1990) The Symbol Grounding Problem. In *Physica D* 42:335-346

[16] Hebb, D.O. (1949) The Organization of Behavior. John Wiley and Sons, New York.

[17] Huyck, C. (2000). Modelling Cell Assemblies. *Proceedings of the International Conference on Artificial Intelligence* ISBN: 1-892512-59-9 pp. 891-7

[18] Huyck, C. (2000) A Practical System for Human-Like Parsing. In *Proceedings of the 14th European Conference on Artificial Intelligence* W. Horn (ed.) IOS Press ISSN: 0922-6389

[19] Huyck, C. (submitted) Competition in Cell Assemblies to Resolve Ambiguity. In *Proceedings of the 23rd Annual Conference of the Cognitive Science Society*

[20] Just, Marcel Adam, and Patricia A. Carpenter. (1980). A theory of reading: From eye fixations to comprehension. Psychological Review, 87(4):123-154.

[21] Langacker, Ronald W. (1987). *Foundations of Cognitive Grammar. Vol. 1* Stanford, CA. Stanford University Press.

[22] Lewis, R. (1992) Recent developments in NL-Soar garden path theory. Technical Report CMU-CS-92-141. School of Computer Science, Carnegie Mellon.

[23] Lippmann, R. P. (1987) An Introduction to Computing with Neural Nets. *IEEE ASSP Magazine* April 1987.

[24] Marcus, Mitchell P. 1980 *A Theory of Syntactic Recognition for Natural Language* Cambridge, MA: MIT Press.

[25] Palm, G. (2000) Robust identification of visual shapes enhanced by synchronisation of cortical activity. In *EmerNet: Third International Workshop on Current Computational Architectures Integrating Neural Networks and Neuroscience*. Wermter, S. ed.

[26] Pinker, S. (1987) The bootstrapping problem in language acquisition. In *Mechanisms of Language Acquisition*. MacWinney, B. ed. Erlbaum.

[27] Pinker, S. (1994) The Language Instinct. Penguin Books, London.

[28] Pulvermuller, Friedemann. (1999) Words in the brain's language. In *Behavioral and Brain Sciences* 22 pp. 253-336.

[29] Pulvermuller, Friedemann. (2000) Syntactic Circuits: How Does the Brain Create Serial Order in Sentences? In *Brain and Language* 71 pp. 194-9.

[30] Church, Kenneth W. and Lisa F. Rau. (1995) Commercial Application of Natural Language Processing. In *Communication of ACM* 38:11 pp. 71-80.

[31] Reddy, R., L. Erman, R. Fennell, and R. Neely. (1976) The HEARSAY speech understanding system: An example of the recognition process. In *EIII Transactions on Computers* C-25: 427-431

[32] Roark, B. and M. Johnson (1999) Efficient Probabilistic Top-Down and Left-Corner Parsing. In 37th Meeting of the Association of Computational Linguists

[33] Rochester, N., J. H. Holland, L. H. Haibt, and W. L. Duda (1956) Tests on a Cell Assembly Theory of the Action of the Brain Using a Large Digital Computer. In *IRE Transaction on Information Theory* IT-2, pp. 80-93

[34] Rosch, Eleanor and C. Mervis. (1975) Family Resemblances: Studies in the Internal Structure of Categories. In *Cognitive Psychology* 7 pp. 573-605.

[35] Sakurai, Yoshio. (1998) The search for cell assemblies in the working brain. In *Behavioural Brain Research* 91 pp. 1-13.

[36] Schank, R. and R. Abelson. (1977) *Scripts, Plans, Goals and Understanding.* Hillsdale, NJ: Lawrence Erlbaum.

[37] Schuz, A. (1995) Neuroanatomy in a Computational Perspective. In *The Handbook of Brain Theory and Neural Networks*. Arbib, M. ed. MIT Press pp. 622-626.

[38] Shastri, L. (2000) SMRITI: a computational model of episodic memory formation inspired by the hippocampal system. In *EmerNet: Third International Workshop on Current Computational Architectures Integrating Neural Networks and Neuroscience*. Wermter, S. ed.

[39] Singer, W. (1995) Development and Plasticity of Cortical Processing Architectures. In *Science* 270 pp. 758-64.

[40] Spatz, H. (1996) Hebb's concept of synaptic plasticity of neuronal cell assemblies. In *Behavioural Brain Research* 78 pp. 3-7.

[41] Van Petten, C. and M. Kutas. (1988) Tracking the Time Course of Meaning Activation In *Lexical Ambiguity Resolution*. Small, S., G. Cottrell and M. Tanenhaus eds. Morna Kaufmann 0-934613-50-8

[42] Wickelgren, W. A. (1999) Webs, Cell Assemblies, and Chunking in Neural Nets. *Canadian Journal of Experimental Psychology* 53:1 pp. 118-131

[43] Wittgenstein L. (1953) *Philosophical Investigations* Blackwell, Oxford.