

# How Language Could Have Evolved

Ken Del Signore  
North Aurora, IL, USA\*

*This paper begins to develop a biologically inspired computational model of the Human language faculty and some associated thought processes. This model is developed starting from a simple proto-language, which humans are assumed to have inherited at speciation. This proto-language consists of single symbol exchange using a small set of symbols; similar to the observed dialogue systems in the existing Great Ape families. Computationally, the model is built using a single class with the form of a Markov graph node. Instances of this node class are used to symbolically represent words. The model is built iteratively in main() as a single graph. Nodes are added to this graph using a merge, or conjunctive join, operation between any two existing nodes, notionally labeled as head and copy. A simple first order graph is developed which is hypothesized to be common to all Mammals and to generate shared mammal behaviours. This graph is then extended to allow for more complex human language and thought processes.*

## 1. Introduction

We take the simplest possible communication to be a single symbol, exchanged from one communicator to another. As an example, consider a hiker that is lost in the woods and builds a pile of rocks, and then moves on. If a second hiker subsequently finds the rocks, then we can say that a single symbol has been exchanged. The information conveyed is the same as if the first hiker had just stood next to the second and said the adverb “here”, except that the hiker said “here” some long time prior and the symbol (the rocks) held the information through time. There is no other temporal or spatial information conveyed.

To modify this example, if while the hiker is piling the rocks, s/he hears the second hiker coming straight on and calls out “here”, then the second hiker will have, at the simplest, one additional quanta of information, namely a measured value of the distance. This second quanta is analogous to a floating point variable that can take on a continuous value.

Single symbol dialogue systems are observed throughout the animal and plant kingdoms. The symbol “here” is often the exchanged symbol in these systems. A flower can be interpreted as a single symbol, exchanged between a plant and its pollinators, with the meaning of the flower being “here” (Chomsky 2015). A symbol is defined to contain two quanta of information: label and value; with value being possibly zero or unspecified. In the examples above, the second communicator measures the value locally to itself using the externalization made by the first communicator.

---

\* kendelsignore@gmail.com

The neocortex gives mammals the ability to store and recall sequences of symbols with relative ease. This ability gives mammals considerably more complex behavior relative to non mammals. Mammals can input and store sequences of symbols in combinations never experienced before and later recall and utilize this information; an example of which would be the second hiker remembering the path out of the woods and walking out with the first hiker.

All known mammal dialogue, excluding human, uses or can be easily reduced to single symbol exchange. The great apes have possibly the most developed system; using a gestural vocabulary of approximately 80 gestures to convey 15 unique meanings as commands and questions (Bryne 2017). The meanings loosely correspond to the hypernym forms of various parts of speech categories [ here, no, give, on, on?, play? ], which are discussed further below.

Among the Hominins, stone tools provide the first evidence for advancement in behavior. The initial Mode I tools date to 3.3 Mya and remained at a relatively fixed level of design and refinement for over a million and a half years. The early hominins did not evidently undergo much generational change; contrary to the current human cliché “kids these days...”. Mode II tools were then developed and these spread slowly throughout the existing hominin range over the next million years. When humans first speciated, they inherited a Mode II toolkit. They had animal hides, cord, hafted hand-axes, spears, and cooked meals, to name a few of their initial conveniences. Hominins had been hunting elephants and hippopotamuses since at least 400 Kya and early human sites dated 200 Kya also contain evidence that they subsisted on these animals.

Humans then began making rapid advances to their toolkits and left Africa approximately 75 Kya. The sudden change to the rate of change of the toolkit just prior to leaving Africa is suggestive that, at the simplest, a single change could have taken place in the hominins to allow them to make these advances. Our current human language (faculty) is argued to be this change (Bolhuis 2014). The use of complex sentences would presumably have allowed the hominins to more easily accumulate knowledge and transfer it to each other and their children. Daily storage and recall of unique sequences would also permit hominins the ability to mentally reconstruct scenarios after they occurred, which would allow them to explain them to others and explore possible solutions when time permitted.

All available evidence indicates that the current human language faculty and cognitive functionality was completely formed before humans left Africa and that it hasn't changed since (Bolhuis 2014); which would be a signature of the single change in Africa hypothesis. The argument for this is that babies from any culture can grow up in any new culture and will readily acquire the new culture and language, which is taken to imply that no changes to the human language faculty or other cognitive functions have occurred in humans since we left Africa.

For the present inquiry, the two main historical developments of interest are the mammalian neocortex and the human toolkit change. The neocortex is viewed as having endowed mammals with the ability to conceptualize symbols, to form vocabularies of these symbols, and to input, store, recall, and utilize random sequences of these symbols. The human change is viewed as happening when the neocortex became large enough to support a new thought process and/or a new thought process was developed.

## Sections

2. The Interface
3. Short term memory, (stm) sequences and recall
4. Protolanguage, mammalian single symbol exchange
5. Movement, boys eat what? what boys eat?
6. Adverb periodicity, here I now eat daily
7. Compare function at the merge, here... HERE
8. Conjunctions, illicit conjunctions, movement+conjunction
9. Long term memory, sleep, prediction
10. Verbs and prepositions
11. Past and future, progressive and perfected, singular/plural
12. Executive function in main, input-predict-[respond], ponder-[respond], sleep
13. Summary

## 2. The Interface

We begin with one of the main minimalist assumptions of Linguistics, namely that communicators have some common internal neurological/symbolic representation of each word in their vocabulary. This is shown as the two blue “interface parcels” in the diagram below, which borrows from similar diagrams in (Pinker 1999 and Berwick 2013). In this example, while walking out of the woods, the two hikers roust a duck, which causes the duck symbol to activate in each hiker’s interface parcel. We assume that each hiker processes their unique neural input of the event and each conceptualizes, or activates, an internal symbol that corresponds to “duck”. Also note that if one of the hikers is replaced by another mammal, such as a dog, the interface parcel representation would still be valid.

For our initial purposes we assume that some analogous neural parcel exists that contains many neural attractor states that represent words or symbols. Such a neural parcel can be thought of as analogous to a two dimensional optical character recognition neural network, wherein each character is a unique attractor state of the OCR neural network. Each character corresponds to a subset of nodes in the 2d array that fire and lock into the active state when the array is input a noisy bitmap of a scanned character and allowed to run freely into the nearest attractor state.

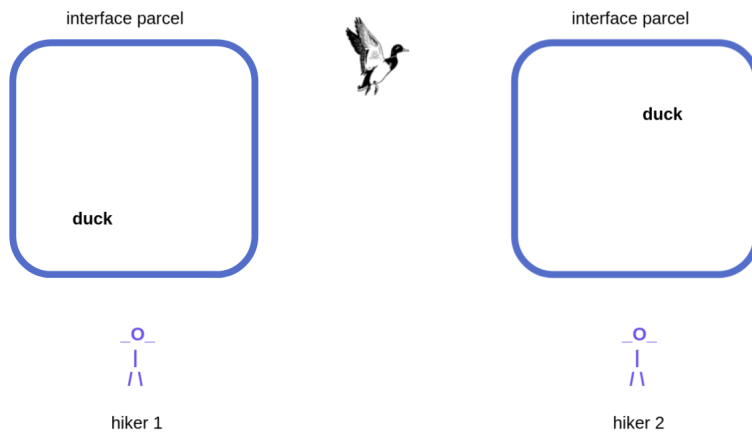
The symbols in this interface parcel, once formed, are assumed capable of subsequent re-activation on similar input. Furthermore, once activated, we assume that the symbols retain this state information for a short time and are more easily re-activated (Hubel 1980). This is analogous to the human ability to be given a random word and then be able to repeat that word on command some short time later.

The interface parcel can be abstracted to represent all of the symbols that we are physically able to internalize and externalize. The temporal sequence of all such symbols can be thought of as our stream of consciousness.

## 3. Short term memory, (stm) sequences and recall

Behaviours involving short term memory are easily observable in mammals and many other animals. In humans, we can easily form stm associations between any two randomly picked symbols in our vocabularies.

As an example, we can extend the duck scenario above such that the surprise of the duck rousting causes the first hiker to sneeze. Following this, the second hiker’s

**Figure 1**

The roosting of the duck provides unique sensory input to each hiker's interface parcel, which we assume causes the "duck" symbol to fire (or conceptualize) therein. This is posited to occur as some subset of the nodes in the interface parcel firing into a stable attractor state.

interface parcel would contain the activated symbols "duck" and "sneeze" and these would be stm bound such that if some short while later a second duck was rousted, then this would cause the second hiker to recall the sneeze symbol and to expect the first hiker to sneeze again. We can say that the second duck caused the hiker to "think of" the sneeze symbol.

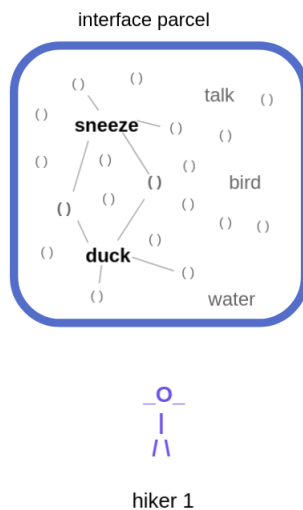
Such a random two symbol stm mechanism can be built by the current model using the assumed node functionality. We (hypothetically) introduce many additional nodes to the interface parcel, referred to as stm nodes. These stm nodes are assumed to be equivalent to the symbol nodes except that they are unlabeled. The stm nodes are assumed to be randomly and sparsely connected to the labeled symbol nodes (Hawkins 2005).

When two random symbol nodes, such as duck and sneeze, are activated and fire, they each provide input activation to their respective stm nodes. If a subset of these stm nodes is common to both symbols, this subset can become activated over background due to having 2x more input activation than the stm nodes that are not common. The elevated input level is then assumed to persist for some time interval, allowing for subsequent short term associative recall.

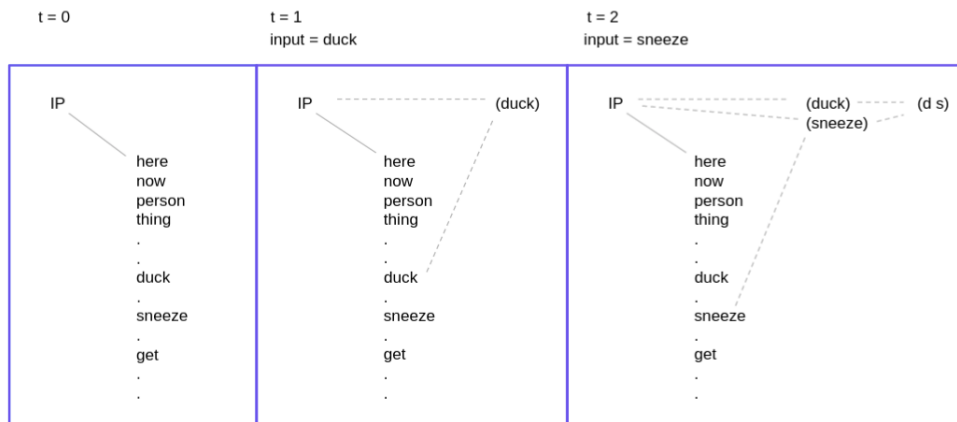
Computationally, stm memory can be implemented as a single node created by a merge function between a root node "ip" (interface parcel) and its child nodes as shown below. The (stm) nodes are created by the merge function that runs each time a node fires. Bidirectional connectivity is assumed possible in all connections.

The merge operation can be applied recursively between other recent stm nodes to create the (d s) node above right. This node then stores the short term association between duck and sneeze.

The stm nodes are created at run time using a merge constructor function of the Node class as shown below. This function takes two nodes as input, labeled "head" and "copy". Bidirectional links are set up between the head and copy nodes in the merge constructor function.

**Figure 2**

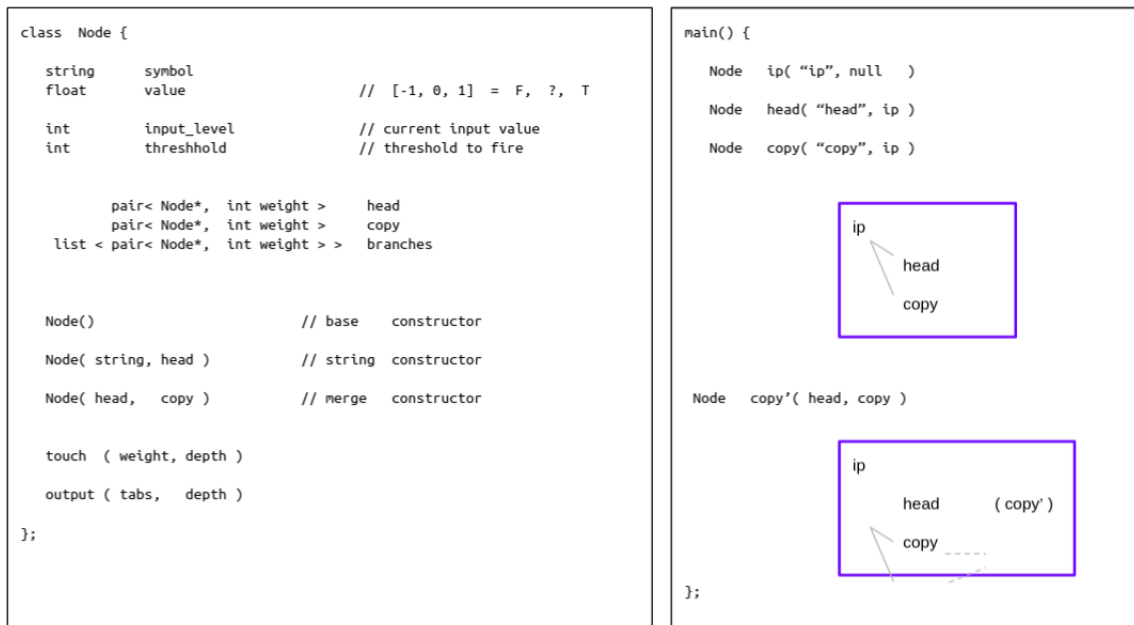
Many sparsely connected (stm) nodes are capable (under appropriate conditions) of forming short term association between any two symbols in the parcel.

**Figure 3**

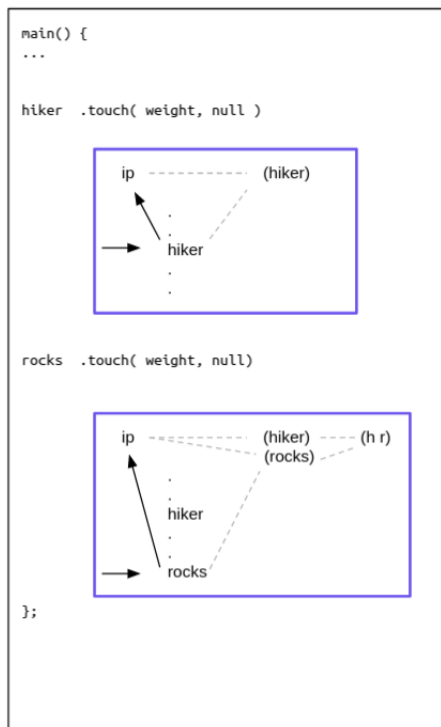
The child nodes fire based on external input and cause then touch and fire the ip head node, The ip node performs the merge between itself and the node that touched it. This creates an (stm) node as shown.

The sequence: "hiker rocks" is input with the "touch" function calls in main() as shown below.

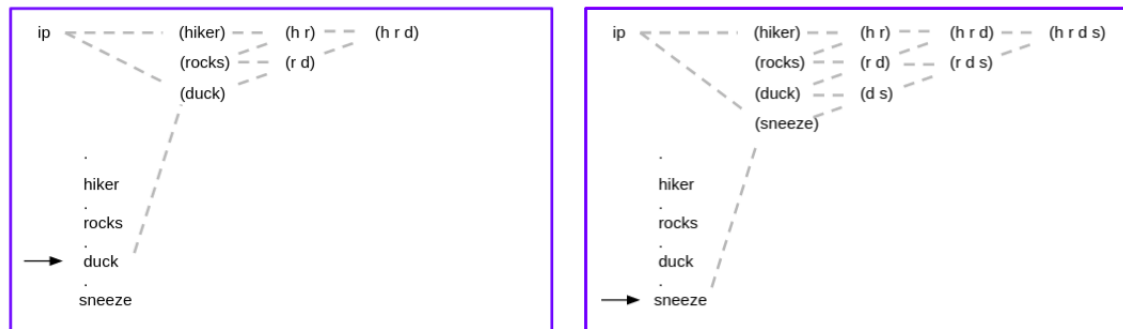
The structure formed in (stm) memory can then be used for output of the hiker, rock, duck, sneeze symbols. Using the recursive algorithm shown in the pseudo code in figure 7, the symbols can be output in the order that they occurred.

**Figure 4**

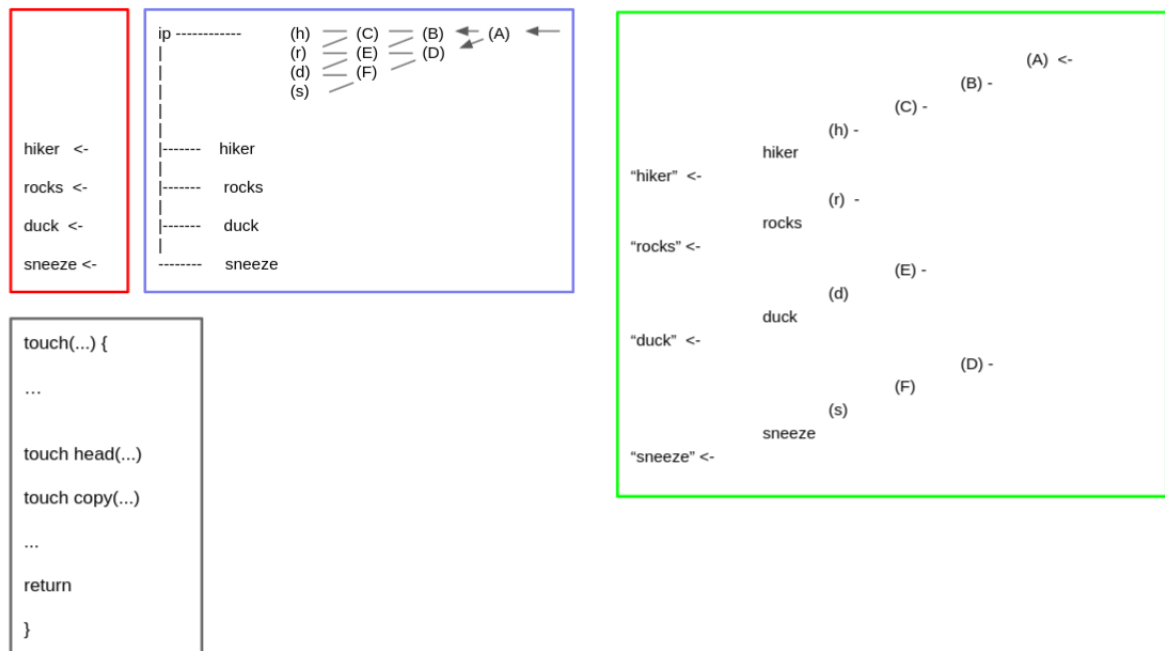
The Node class and the use of the constructor functions in main().

**Figure 5**

The hiker.touch() function call causes the hiker node to fire, which then causes the (head) ip node to fire. The stm merge node is created when the ip node fires.



**Figure 6**  
The stm structure grows iteratively as more symbols are input.



**Figure 7**  
A c++ pointer to the rightmost (stm) node, denoted by "(A)", can be returned to main() and used to output the stored node sequence using the touch() function.

#### 4. Protolanguage, Mammalian single symbol exchange

Following the suggestion of (Chomsky 2015), we assume a simple (truncated english) corpus of the form: [ here now me thing get do go ]. This corpus is drawn from the hypernym forms of the parts of speech: [ adverb noun verb ].

gesture	meaning	part of speech	hypernym form
grab, ...	stop that	negative	stop
mouth stroke, ...	acquire object	verb - get	give
bite, ...	contact (affection)	verb - feel -?	touching
big loud scratch	init grooming	verb - point	do
arm swing, ...	move away	verb - line - negative	go
beckon, ...	move closer	verb - line	come
big loud scratch	travel with me	prep	with
jump, ...	follow me (sex)	prep	with
foot present, ...	climb on me	prep	on
reach	climb on you?	?	on?
present location	groom here	adverb - here	here
leaf clipping, punch ground	sexual attention: male	?	flirt
leaf clipping	female	?	flirt

**Figure 8**  
The chimpanzees exchange approximately 15 unique meanings using gestures. The hypernym forms are mapped to the closest meaning and can all be nominally matched. No grammar or random combinatoric use of gestures is observed. All species of Great Apes share a common set of 100 gestures with each species using a subset of 60 gestures, however, the gesture to meaning mapping is different in every species.

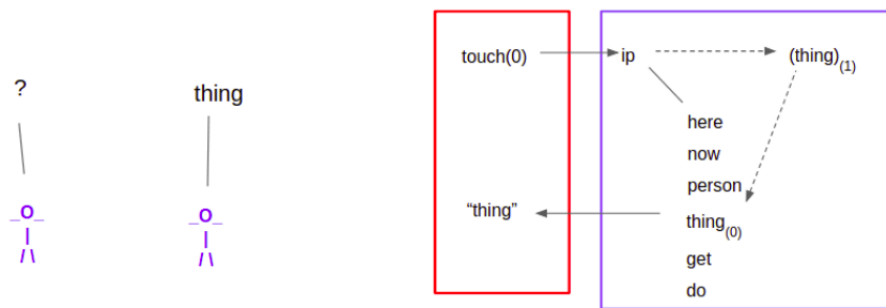


**Figure 9**  
The apes use multiple gestures per meaning and young adult apes are sometimes observed to cycle through these as a repetitive sequence of gestures, suggestive of a hypernym-hyponym structure (Byrne 2017).

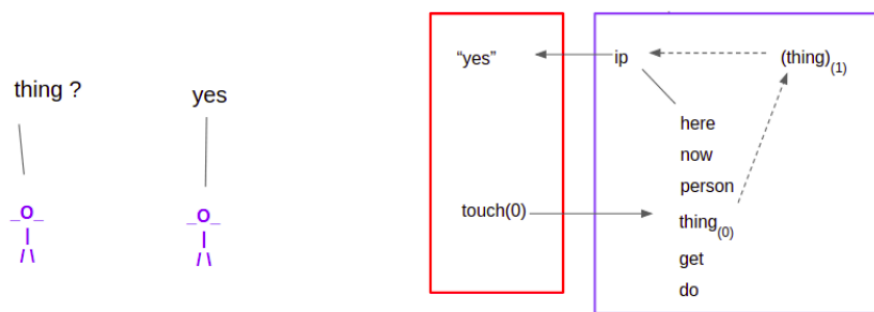
This initial corpus is similar to the reported meanings exchanged in chimpanzee gestural dialogues. The table below is drawn from the meanings derived from a large video corpus of wild chimpanzee single symbol dialogue exchanges (Hoabaiter 2014). The meanings are mapped to a part of speech and then to a hypernym word for that POS. The verb POS forms labeled ‘point’ and ‘line’ correspond to [ do go ].

Switching back to humans, our understanding of the neocortex is expanding at a great rate using many types of experimental methods. fMRI experiments in (Epsteint 1998) and (Huth 2013) have identified two voxels ( <2mm3 ) that fire in response to words that are hyponyms of (person/place) and (thing). Movies are shown to volun-



**Figure 10**

The value of zero passed in the touch(0) function indicates a question; -1, 0, 1 = [ no, ?, yes]. The simplest input form of a question is the function call: ip.touch(0). The stm (thing) node is assumed to have been previously formed.

**Figure 11**

All symbols can be input as questions with the touch(0) function call as: thing.touch(0) .

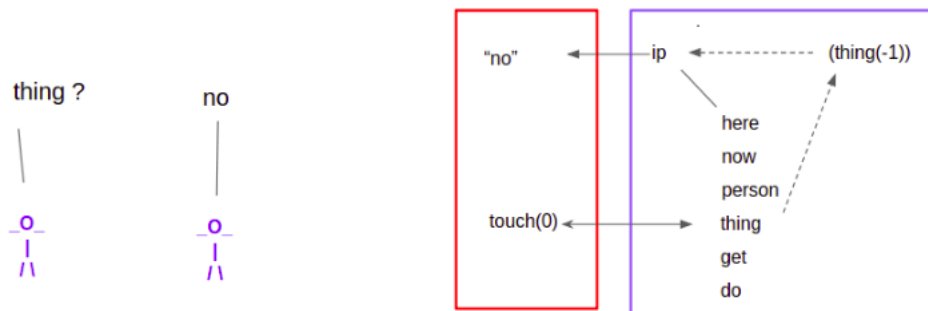
teers and the hypernym mappings from WordNet are used to tag 1800 nouns from the movie dialogues to person, place, or thing. In all volunteers, these two voxels can be identified in similar locations on a neocortex flatmap and show activation when the corresponding hyponym words are used in the movie dialogues.

The symbols of the proto-language are assumed to be formed as child nodes in the interface parcel as shown. The stm memory allows for storage of state information and for simple dialogues.

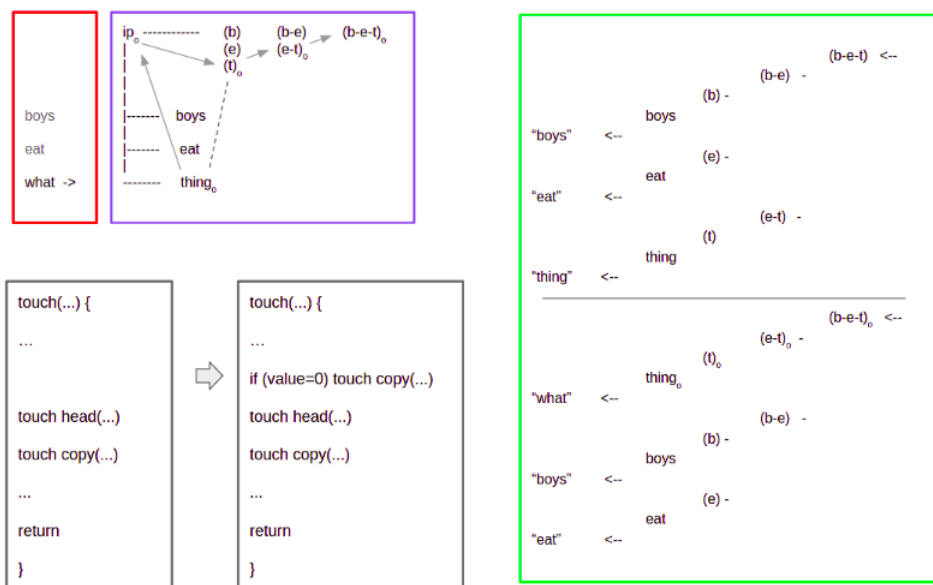
### 5. Movement, boys eat what? what boys eat?

A mechanism for movement can be implemented by using the stored values in the (stm) nodes and modifying the pseudocode as shown. The input value of zero is propagated to the (stm) nodes as shown.

The output of the c++ prototype is shown in Figure 14. The input sequence is “boys eat what”, where what = thing:0. Following input, a pointer the the (b e t) node is returned to main() and this is used in the call: (b e t)->touch() to invoke the output: “what boys eat”

**Figure 12**

An stm node having a value = -1 indicates negation. In this scenario, (thing(-1)) was previously created via an stm merge.

**Figure 13**

The single line modification to the pseudocode will alter the output ordering from "boys eat what" to "what boys eat".

```

kwd1:code20$ c++ graph1.cpp
kwd1:code20$ ./a.out
-----
IP
    boys 0
    thing 0
    eat 0
-----
IP:1 <--
|
(b)
    boys:1 <--
    eat:1 <--
IP:1 <--
|
(e) (b e)
    thing:0 <--
IP:0 <--
|
(t) (e t) (b e t)
-----
IP          (b) 1      (b e) 1      (b e t) 0
            (e) 1      (e t) 0
            (t) 0

            boys 1
            thing 0
            eat 1
-----
time interval occurs

                                (b e t):0 <--
                                (e t):0 <--
thing:0 <--      (t):0 <--
                                (b e):1 <--
boys:1 <--      (b):1 <--
                                (e):1 <--
eat:1 <--

```

**Figure 14**

c++ prototype output of “what boys eat?” Here “thing:0” is defined to be “what”

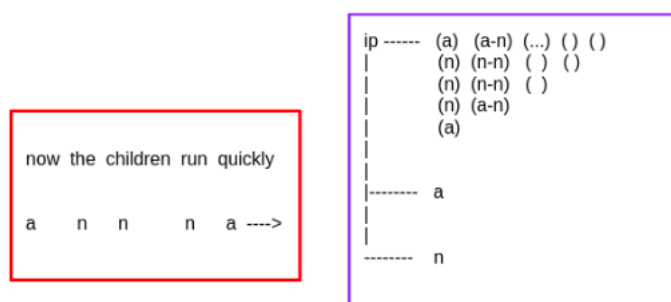
## 6. Adverb periodicity, here I now eat daily

The set of words in a language can be bifurcated and mapped into two symbols “a” or “n”, which corresponding to the adverbs/adjectives and the nouns/verbs/prepositions

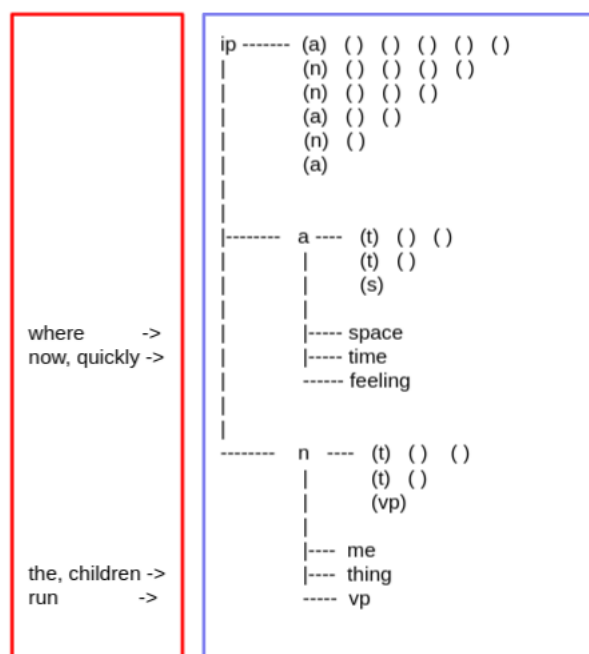
The Wordnet English corpus of 120K definitions and 60K glossary sentences can then be encoded into such reduced sequences and used as input as shown. These sequences can be considered a form of “truncated english”.

Oscillation of the a-n graph is hypothesized to produce adverb periodicity in the spoken languages.

The terminal nodes of the ip graph are then bifurcated further as shown.

**Figure 15**

The 120K unique words in the WordNet corpus can be mapped to 'a' or 'n'.

**Figure 16**

The a and n branches bifurcate further. The words in the corpus are mapped to one of the six hypernym forms. The stm memory mechanism functions within each branch of the graph. "now the children quickly run where?" is input and stored as "time thing time vp space?"

## 7. Compare function at the merge

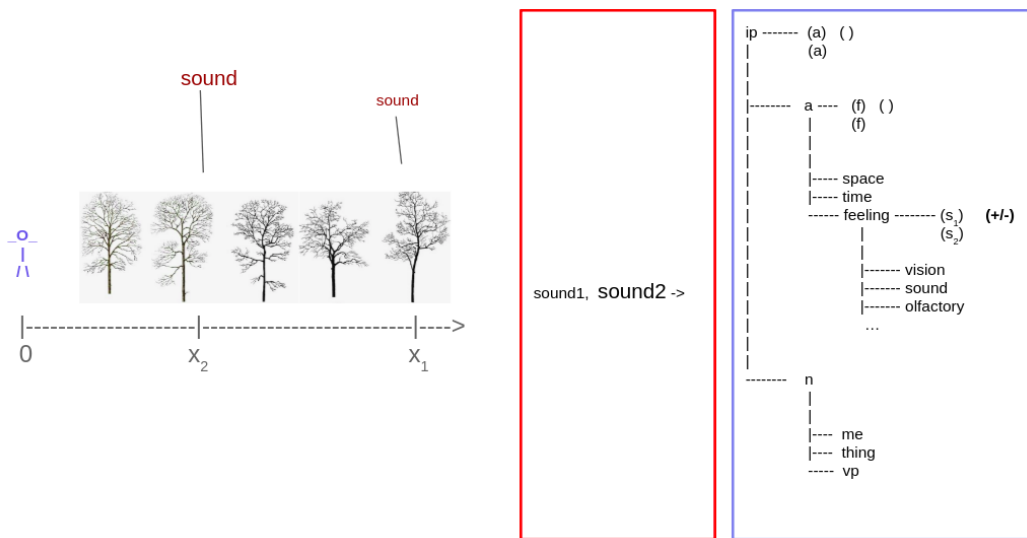
here -here -HERE -HERE

A compare mechanism exists such that similar sensations, separated in time, can be compared. A familiar example is hearing the first two intermittent sounds of crunching leaves when someone or something is moving in the woods, relative to a fixed observer.

The change in intensity is available at the interface parcel as an internal feedback to the observer.

The ability to compare two of the same sensory inputs separated in time is an important evolutionary advantage to all animals. The initial measured information from each sensation must be stored through time and then compared with the second measurement at a later time. This functionality can be achieved at the second level (stm) merge, shown in the adverb branch of figure 17, using the values of the two parent nodes.

This function, storing a value and using it in a compare operation later in time, is similar to that of the Reichardt model used by Hubel to explain directionally sensitive neural circuits in V1 (Hubel 1980).

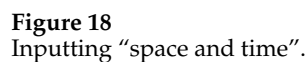


**Figure 17**

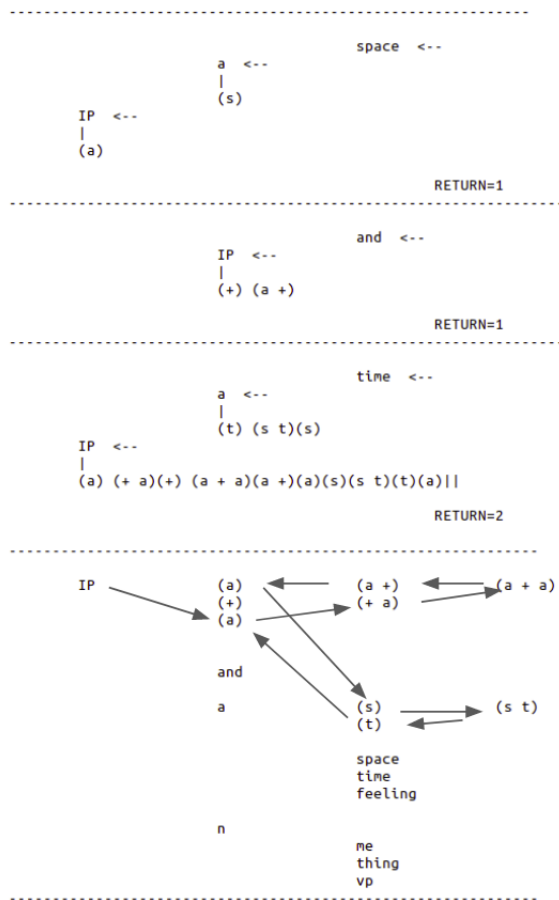
State information is stored in the adverb branch and can be used in a subsequent compare operation.

## 8. Conjunctions, illicit conjunctions, movement of conjunctions

An “and” node is added to the IP graph, with the corresponding (stm) symbol labeled (+), as shown below. As introduced, this node would have no additional properties not already described.



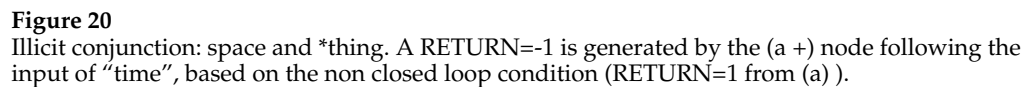
The return value of 2 can then be detected in the touch function and used to trigger a Hebbian enhancement of the weights between the calling and called nodes in (stm) memory.

**Figure 19**

Closed loop detection and Hebbian one shot binding. The touch() function builds a list of node pointers as it recursively calls itself and can detect the closed loop and return an enhanced return value.

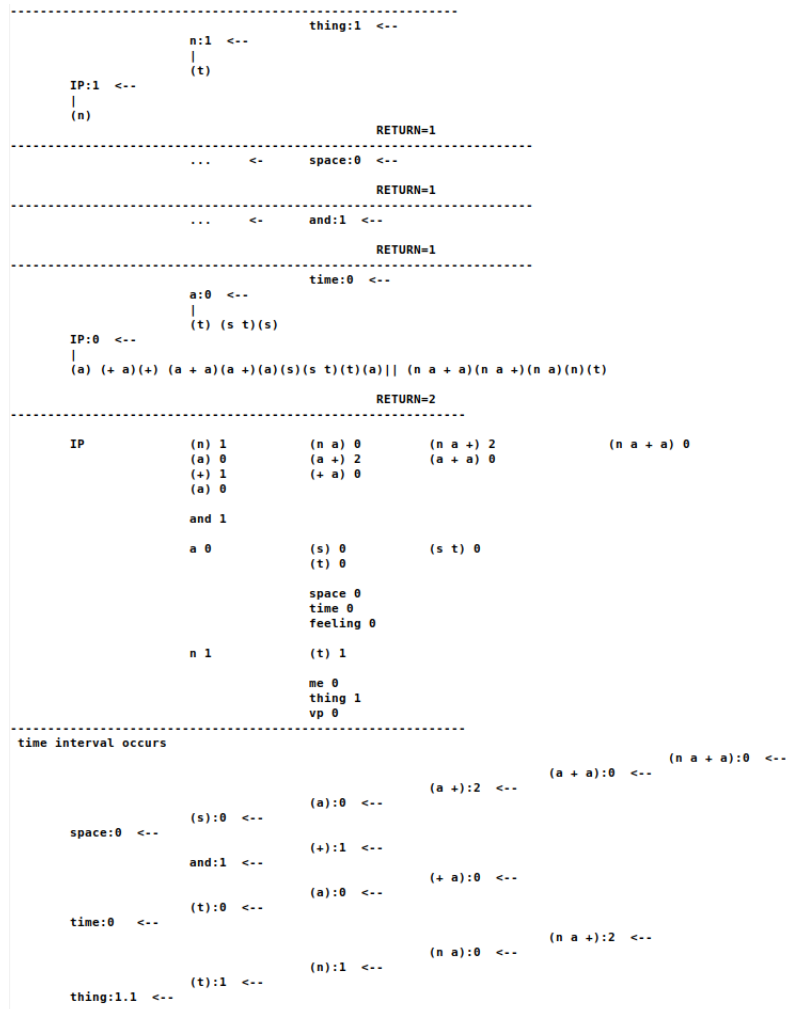
An illicit conjunction sequence can be created with the model by introducing a single logical change to the (stm) nodes that are copy-rooted by the (+) node, namely, that if the return value in the Hebb loop is <2, then return -1. This is shown in Figure 20, where the (a +) node would detect the non closed loop condition (RETURN=1) and change the return value to -1.

The value=-1 is retained in the (stm) structure and can be propagated to subsequent (stm) recall. Such a mechanism could be used during sleep to exclude the stored (stm) sequence from normal [ltm] sequence storage.



16

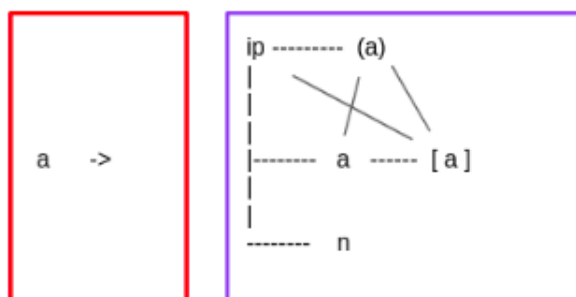




**Figure 21**  
movement of a conjunction is accomplished using the modified bindings and the value of zero stored in IP (stm) memory.

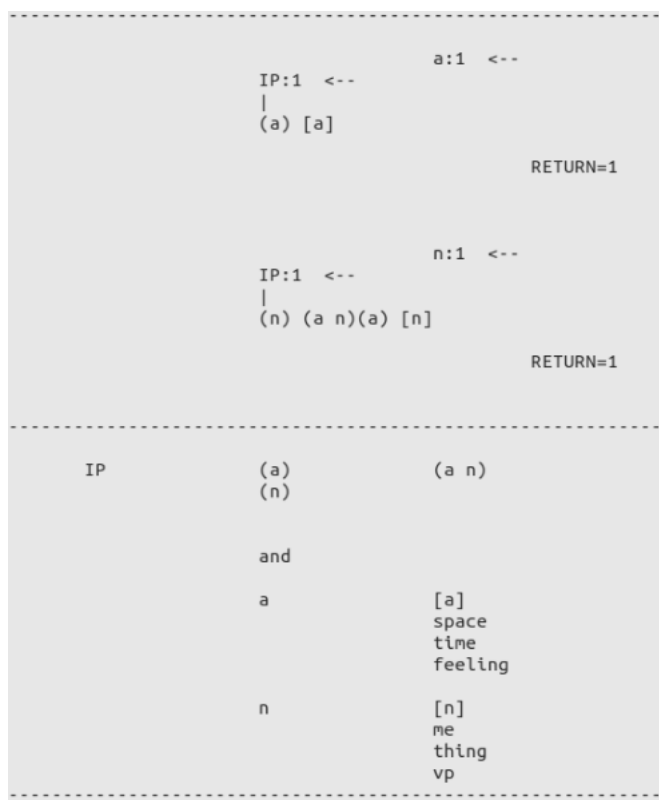
### 9. Long term memory, [ltm] = merge( child, head ), Sleep, Prediction

17

**Figure 22**

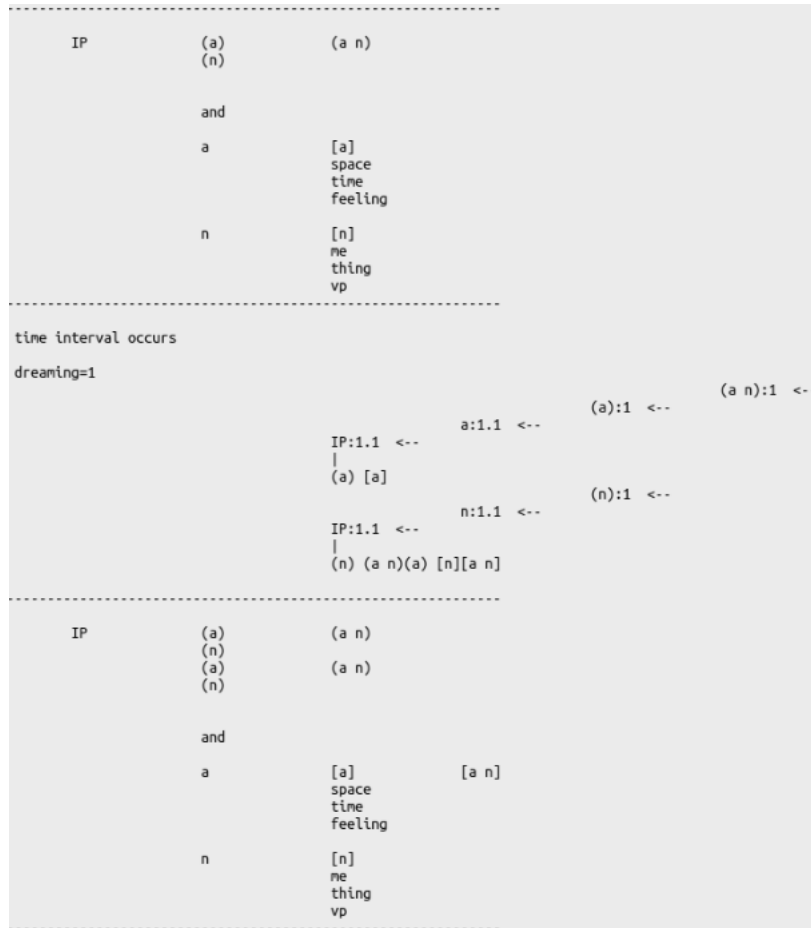
A long term memory node is formed by a reverse merge operation.

In the c++ prototype, the creation of [ltm] nodes by the merge() function can be toggled on and off with a single global variable. Once created, [ltm] nodes can be called by the (stm) nodes at the end of the existing touch() function using the link that was created at merge() time. This input sequence is shown in Figure 23 below. The call to the [ltm] node can also be toggled on and off; which would be analogous to inputting into (stm) rapidly without time to process the touch() call to the [ltm] node.

**Figure 23**

The [ltm] nodes fire at the end of the existing (stm) node sequences.

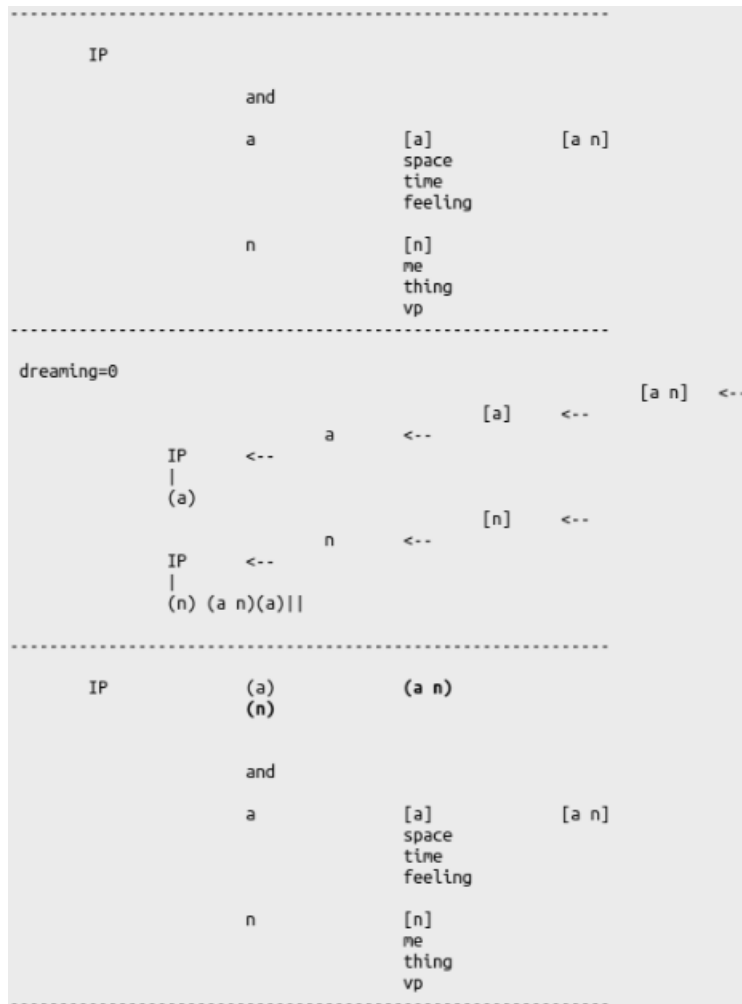
A dreaming mode is introduced in main() to replay the accumulated (stm) sequences and to build the associated sequences in [ltm] memory. This allows the [a n] node to be created as shown in figure 17.



**Figure 24**

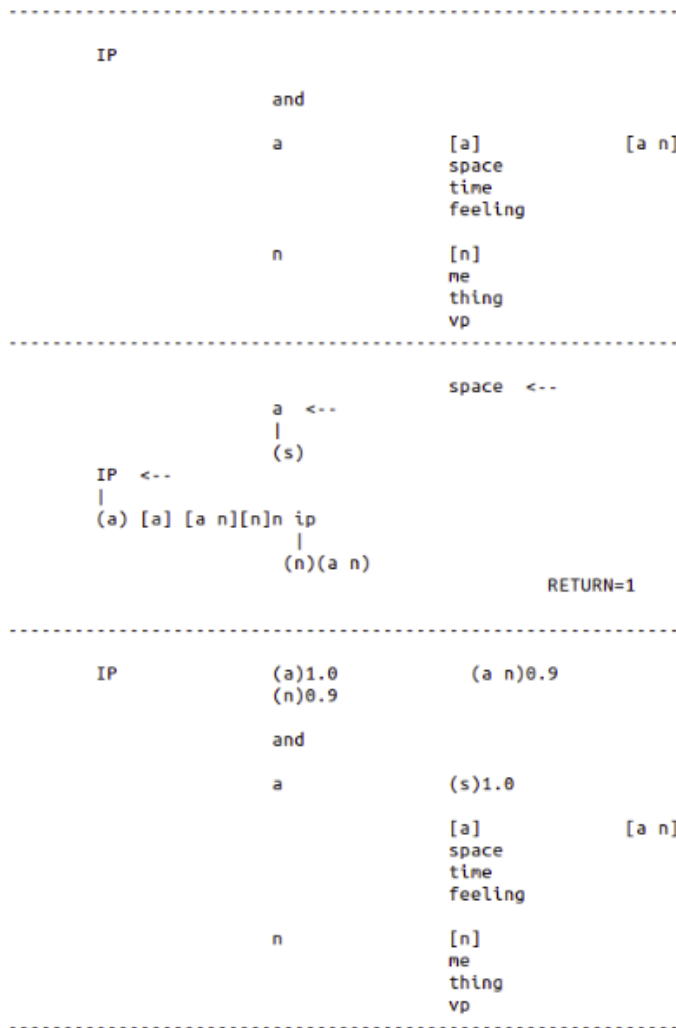
A dreaming mode allows previously created stm memory structures to duplicate in [ltm] memory by replaying the stored (stm) pattern with modified network parameters for the merge function.

Once [ltm] nodes have been created, the (stm) nodes can be cleared from the graph and the [a n] node can be used to reproduce the node firing sequence that will rebuild the (stm) pattern in memory.

**Figure 25**

The [a n] node can then be used to recall the (a) (n) pattern to (stm) memory.

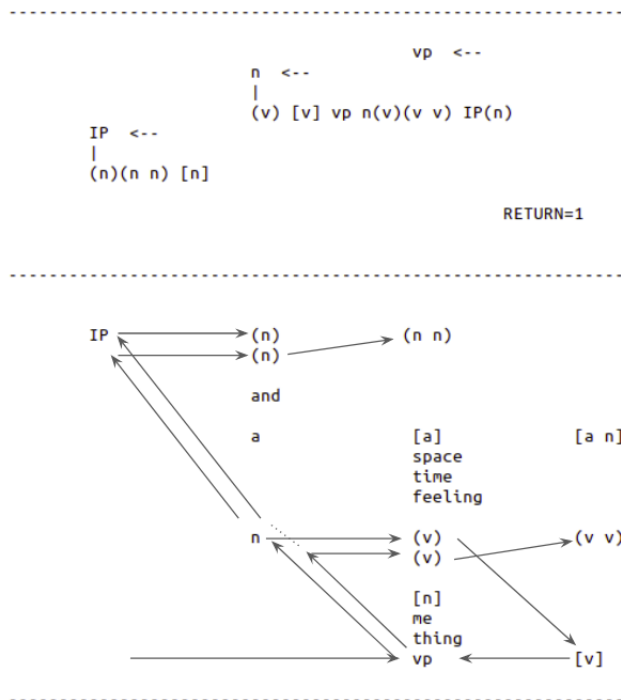
A simple prediction process can occur using the stored [a n] sequence, namely to predict an (n) node in (stm) memory following input of an “a” symbol. This is shown in the graph flow below, where the symbol “space” is input, leading to the [a] -> [a n] -> [n] -> n -> IP -> (n).9 -> (a n).9 sequence to fire. Here the predicted (stm) nodes are created with value = .9 to differentiate them.



**Figure 26**  
Prediction of "n" after the input of "space"

## 10. Verbs and prepositions

The the [vp] node is manually modified in main() to cause its head node (the vp node) to be touched when it fires as part of the existing vp node firing sequence. This causes a second layer of (stm) nodes to be built in (stm) memory as shown below. This second (stm) layer can then alter subsequent graph operations, similarly to the "and" node.

**Figure 27**

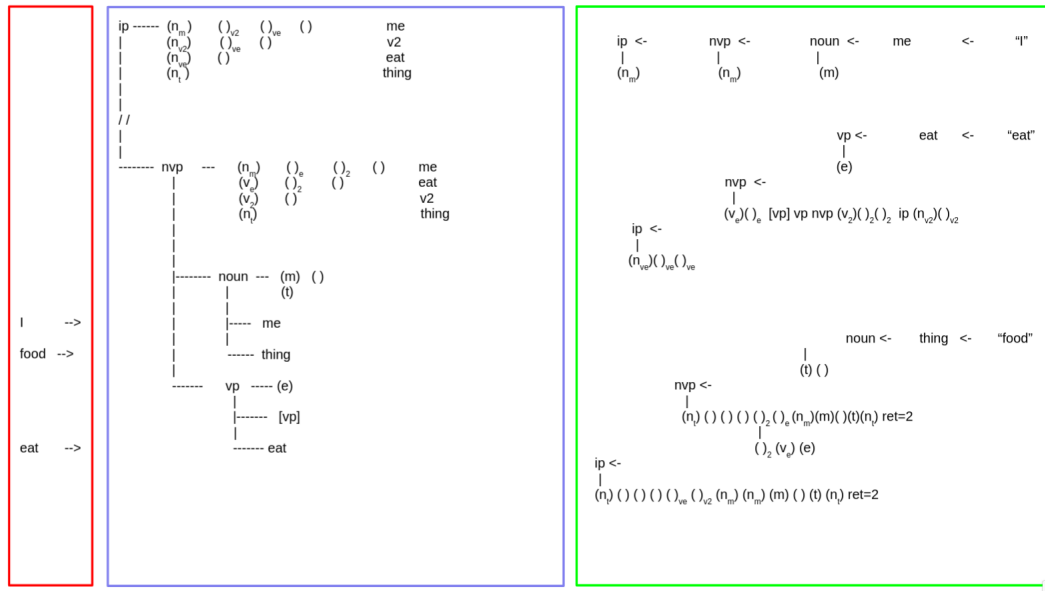
The vp node is the hypernym node for the verbs and prepositions. The weights of the [vp] node are such that it fires the vp node a second time, generating a second loop in (stm) memory. This second loop implements functions of the direct object.

In the figure below, "I eat food" is input, causing the (stm) layers to be built as shown. Input of "food" causes a closed loop to form (return=2) in the nvp and IP (stm) memory. This would permit enhanced binding of the subject and object, using the same mechanism as the conjunction node.

We introduce a modification to the touch() function for the layer of (stm) nodes introduced by the second [vp] loop. This is shown in the diagram below following the input of "food"; a closed loop is detected in the nvp (stm) memory and a return value = 2 is returned to the calling nodes. This is posited to cause the  $()_2$  (stm) node to touch() its copy node when the return value = 2 from the closed loop. This causes the graph flow to follow the (stm) memory back to the specific verb node and would allow for one shot Hebbian enhancement of the (stm) bindings between the verb and the subject+object.

Illicit direct object constructions can be detected to cause a return value of -1. A purpose of this function would be to keep the illicit input node from forming enhanced bindings in (stm) memory by inhibiting the Hebbian one shot weight enhancement mechanism. This function can be implemented using the same mechanism as the conjunction node. In the graph flow shown upper right in the figure below, the input "I eat \*drink" is depicted. Input of "drink" causes the  $()_{v2}$  node to return -1 by the same rule as used in the conjunction  $()_+$  node, namely if the return value (from the  $()_{ve}$  node) is  $< 2$ , return -1.

This similar function between the  $()_{v2}$  and  $()_+$  nodes indicates that this function could be latent in all (stm) nodes and enabled by change of a local variable. This



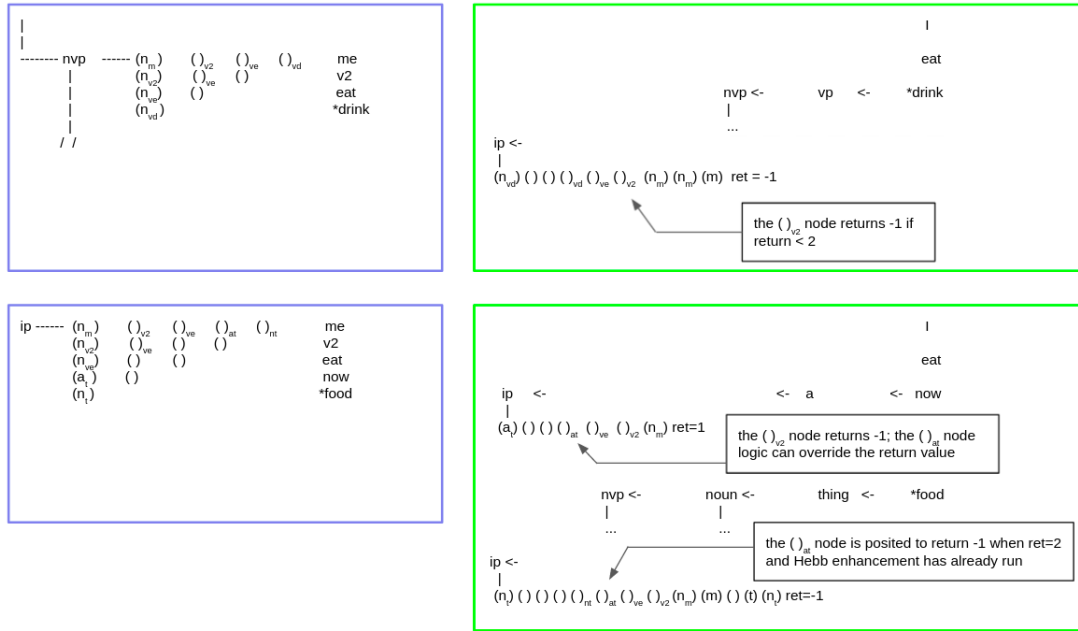
**Figure 28**  
I eat food.

information would be stored in the permanent [vp] and "and" nodes and passed to the (stm) nodes at run time.

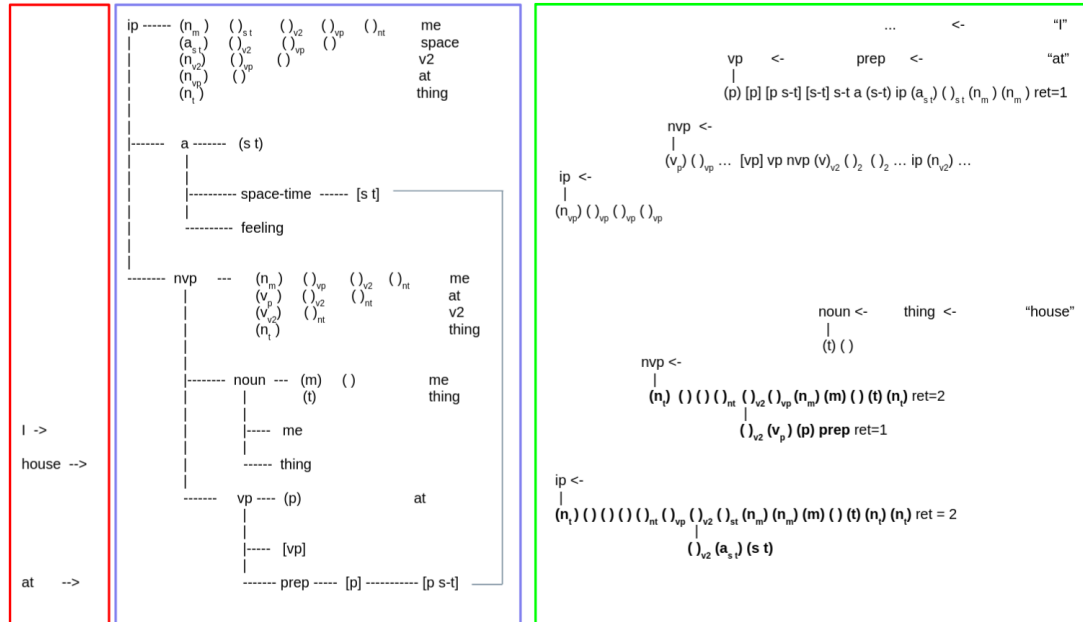
The second construction shown is "I eat now \*food". Following input of the "now" symbol, the ( )<sub>v2</sub> node would return -1 on the existing logic, however "I eat now" is a valid construction. This indicates the ( )<sub>at</sub> node must override the return value = -1, as indicated in the graph flow. However, following the input of "food", the ( )<sub>at</sub> node must produce a return value of -1. This logic could be based on the condition of (return value = 2) + (the Hebbian enhancement already run).

Prepositions are posited to be children of a single node "prep" under the vp node. A manual merge is done from main() between the [p] node and the [s t] node to allow an additional layer of (stm) nodes to form in the ip node's (stm) memory as shown in figure 30. This would allow the Hebbian one shot enhancement mechanism to run onto the "a" branch when "house" is input as shown.

Here we have combined space and time into a single node. This follows the pattern where the branches of the graph branch into two child branches (excluding the verbs).

**Figure 29**

Illicit direct object constructions, "I eat \*drink" and "I eat now \*food".

**Figure 30**

A "prep" node is created in main() with a merge([p], [s t]). The weights of the head and copy nodes are set so as to produce the (stm) layer in IP memory.

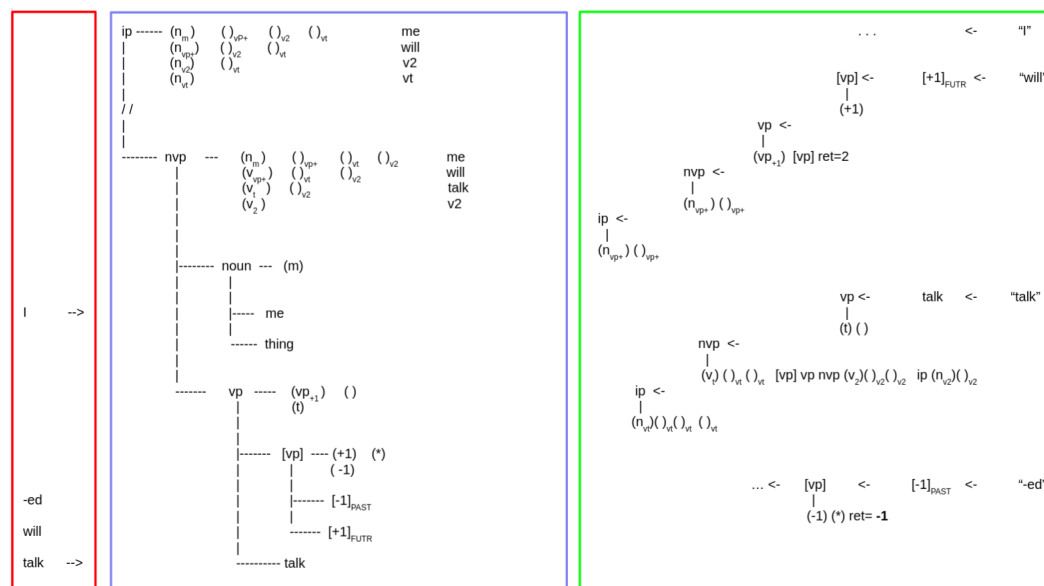


## 11. Past and future, progressive and perfected, singular and plural

This value based mechanism allows for illicit grammar detection in all three categories using the same mechanism, as described below.

We modify the graph to add two child nodes to the [vp] node as shown below. These nodes are assumed to carry default values of +1, corresponding to the past and future tenses. The “will” and “ed” tense markers are mapped to either node as shown. Upon input, the (stm) memory nodes are assumed to carry the +1 value, which permits detection of the illicit construction when the merge function creates the (\*) node.

The future modal verbs can be modeled in sequence, [+6]<sub>can</sub>, [+8]<sub>shall</sub>, [+9]<sub>must</sub>, [+1.0]<sub>will</sub>.



The vp frame node [vp] bifurcates on value to form the past and future tense symbols. Illicit tense detection (I will talk \*ed) is possible at the merge of the (+1) and (-1) child nodes of the [vp] node.

This model implies that at some earlier time the irregular verb “ate” did not have an irregular form and that something caused the usage of the irregular form to be adopted.

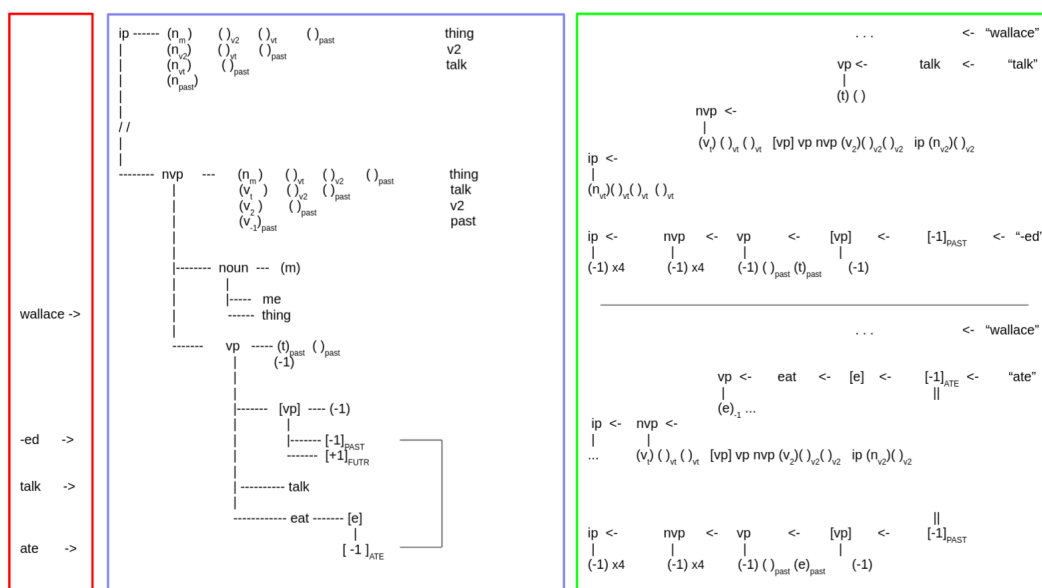


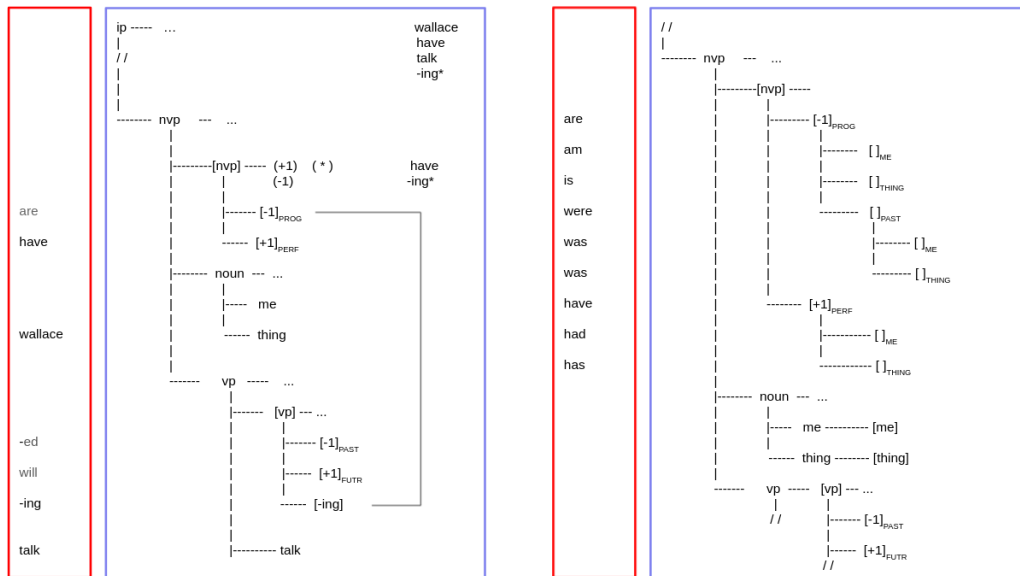
Figure 32

## 11.2 Progressive and perfected

The progressive and perfected tense symbols “are” and “have” are implemented on the graph using the same +1 branching mechanism from the [nvp] node, which similarly permits detection of illicit tense grammar.

The [-1]<sub>PROG</sub> node is posited to merge with the [vp] node to create the [-ing] node. This permits a similar verb agreement check.

The [-1]<sub>PROG</sub> and [+1]<sub>PERF</sub> nodes can then form child nodes by merging with the [-1]<sub>PAST</sub>, [me], and [thing] nodes as shown.

**Figure 33**

Progressive and perfected nodes develop as children of the [nvp] node. A negative signal can be generated by the merge in the [nvp] node's (stm), as in wallace have talk ing\*.

### 11.3 Singular/plural, my, the, a

Singular and Plural nodes are posited to form as value based branches of the [noun] node as shown in figure 34. "my" and "the" are posited to form as merge nodes between the [me]/[thing] nodes and the [noun] node. The [a] node is posited to form as a merge between the [the] node and the [+1]<sub>SINGULAR</sub> node.

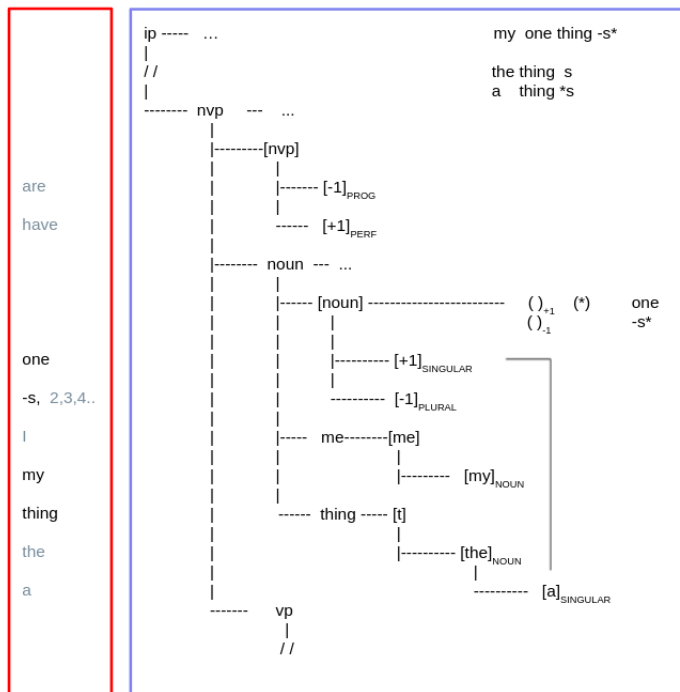
## 12. Executive function in main(), input-predict-[respond], ponder-[respond], sleep

Input of a sentence sequence in main proceeds as a series of calls to the touch() function of each node in the sequence. The touch() function calls cause (stm) memory structures to accumulate in the ip graph, possibly forming enhanced Hebbian bindings, and then return a +1, depending on whether any false signals were generated.

The (stm) memory structures that are built for each sentence can be probed from main() to return the rightmost (stm) node. These (stm) nodes can then be touch()-ed from main() to invoke sequence output.

Following an input sequence, the ip graph can be used to generate a prediction and/or a question based on its stored [ltm] and (stm) memory.

A ponder mode is posited whereby the same prediction mode would run, but with modified graph parameters to allow for deeper recursive function calls to probe [ltm] memory that could be recalled and utilized.



**Figure 34**  
Posited merge nodes created for “the”, “a”, and “my” nodes.

Repeating this process on a training corpus would then accumulate a set of (stm) nodes that can encode the sequences that have been processed since the last sleep cycle. A sleep cycle can then reprocess the (stm) memories into [ltm] structures.

The ip graph is then extended manually in main() by doing merge() function calls between existing terminal nodes, following which, the words in the training corpus are mapped to the new terminal graph nodes and the process of corpus processing is repeated, allowing for additional graph terminal nodes to be added.

### 13. Summary

A symbol is defined to carry a name and a signed value, with value being possibly zero or unspecified.

A symbol’s value is analogous to the spiking rate of a neuron, with the sign being analogous to the relative phase of the spiking signal.

Symbols are assumed to be neurological attractor states formed in an “interface parcel”.

All of our physical senses are assumed to be represented at the interface parcel as symbols that carry name and intensity (value).

The temporal sequence of all symbols that fire in the interface parcel, stimulated either internally or externally, is analogous to our stream of consciousness.

Computationally, the interface parcel symbols are implemented as hyponym (child) nodes of a single hypernym (head) node labeled "ip".

A short term memory model is proposed that operates via a merge operation between the hypernym and hyponym nodes in the ip graph.

The proposed stm merge operation can be applied to other recently created stm nodes to form short term associations between symbols that have recently fired.

A long term memory mechanism is proposed that operates via a reverse merge operation with the head and copy nodes swapped, but with the original copy node's label retained in the newly created [ltm] node.

Long term memory structures are formed in the same manner as (stm) structures. These ltm structures can be recalled to re-fire the symbols (in sequential order) in the interface parcel.

A single symbol dialogue system is assumed using a corpus of basic adverbs, nouns, verbs, and prepositions.

The ip-graph and short term memory mechanism allow for storage and recall of recently acquired information and for simple dialogue sequences to occur.

Questions and commands are implemented by calling the touch(value) function from main() with value = 0, 1.

The ip node takes on the externalizations [ no, ?, yes ] corresponding to ip.value = [-1, 0, 1] .

The (stm) memory mechanism allows any combination of two nodes in the ip graph to be associated together in short term memory.

A graph node labeled "a" is manually added to the ip graph. Symbols that are associated with a value, adverbs, adjectives, feelings, and sensations, map to the "a" branch of the ip graph.

A graph node labeled "n" is added to the ip graph which the remaining non adverb-like symbols map to.

At the simplest, a single oscillation of this graph can bind values stored in the a-branch to the last fired symbol in the n graph.

Oscillation of the a-n graph is hypothesized to produce adverb periodicity in spoken languages.

A conjunction node is added to the ip graph at the first level and operates in a-n (stm) memory as a new symbol type.

A diff() function can be done at stm merge time to generate a (greater than/less than) return value to the interface parcel (to main()).

The direct object node, labeled vp, is manually added to the noun branch and causes an extra ip-graph cycle to run that sets a n-n sequence in ip-(stm) memory.

The verbs and prepositions will form as child nodes to this vp node and use the double ip-cycle mechanism.

A preposition node is posited to develop as a branch of the vp node that causes a second (stm) layer to form from the “a” branch.

The past and future tenses are posited to occur as a value based bifurcation of the [vp] node.

Irregular past tense verbs are formed with a merge between each verb’s [ltm] node and the [ ]<sub>PAST</sub> node.

The progressive and perfected nodes are posited to develop as similar value based bifurcation of the [nvp] node.

The current English grammar develops from these two nodes. The progressive node bifurcates on (merges with) past, me, and thing. Progressive-past bifurcates on me and thing and perfected bifurcates on me and thing.

Singular and Plural are posited to develop from the [noun] node using the same value based bifurcation mechanism.

The “my” and “the” nodes are posited to develop from the [me] and [thing] nodes through a merge with the [noun] node. The “a” node is posited to form through a merge of the [the] and [+1]<sub>SINGULAR</sub> nodes.

#### 14. Acknowledgments

The author has received many helpful comments from many people who have reviewed early versions of this manuscript.

#### 15. References

- Tattersall I., 2016, A tentative framework for the acquisition of language and modern human cognition. In *J Anthropol Sci.* 2016 Jun 20;94:157-66. doi: 10.4436/JASS.94030. Epub 2015 Mar 25. PMID: 27014833.
- Epstein R, Kanwisher, 1998 N. A cortical representation of the local visual environment. *Nature.* 1998;392(6676):598–601. The ‘parahippocampal place area’ (PPA) is described.
- AG Huth, et al, 2013, A Continuous Semantic Space Describes the Representation of Thousands of Object and Action Categories across the Human Brain. In *Neuron.* 2012 Dec 20; In 76(6): 1210–1224. doi: 10.1016/j.neuron.2012.10.014, Figure 2, PPA analysis.
- N. Kruger et al., 2013. "Deep Hierarchies in the Primate Visual Cortex: What Can We Learn for Computer Vision?. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1847-1871, Aug. 2013, doi: 10.1109/TPAMI.2012.272.
- Johan J. Bolhuis, et al, 2014, How Could Language Have Evolved? In *PLOS*, August 26, 2014 <http://dx.doi.org/10.1371/journal.pbio.1001934>

- Noam Chomsky, 2015 Artificial Intelligence, in Navigating a Multispecies World: A Graduate Student Conference on the Species Turn.  
<https://sts.hks.harvard.edu/events/workshops/navigating-a-multispecies-world/>
- Noam Chomsky, 2015, Some Core Contested Concepts. In N. J Psycholinguist Res (2015) 44: 91.  
<https://doi.org/10.1007/s10936-014-9331-5>  
<https://dspace.mit.edu/openaccess-disseminate/1721.1/103525>
- Noam Chomsky, 2016, "Language, Creativity, and the Limits of Understanding" (4-21-16). At <https://www.youtube.com>
- Noam Chomsky, 2011. Language and the Cognitive Science Revolution(s), lecture given at Carleton University 2011 <https://chomsky.info/20110408/>
- Steven Pinker, 1999, Words and Rules. At <https://stevenpinker.com/files/pinker/files/edinburgh.pdf>, figure 1.
- Berwick, et al, 2013, Evolution, brain, and the nature of language. In Trends Cogn Sci . 2013 Feb;17(2):89-98. doi: 10.1016/j.tics.2012.12.002. Epub 2013 Jan 9. See figure 1.
- Collin Phillips, 2003, Linear Order and Constituency. In Linguistic Inquiry Vol. 34, No. 1 (Winter, 2003), pp. 37-90
- Collin Phillips, 2013, Derivational Order in Syntax: Evidence and Architectural Consequences. In Studies in Linguistics 6: 11-47.
- Fitch, 2017, Empirical approaches to the study of language evolution. In Psychon Bull Rev . 2017 Feb;24(1):3-33. doi: 10.3758/s13423-017-1236-5.
- Minsky, 1974, A Framework for Representing Knowledge, Marvin Minsky, MIT-AI Laboratory Memo 306, June, 1974.  
<http://web.media.mit.edu/~minsky/papers/Frames/frames.html>
- Hubel, 1980, Eye Brain and Vision, David Hubel 1980. At <http://hubel.med.harvard.edu/index.html>
- Hubel, D. H., 1957 Tungsten Microelectrode for Recording from Single Units. In Science 1957 Mar 22;125(3247):549-50. doi: 10.1126/science.125.3247.549.
- Hubel, D. H., Wiesel, T. N., 1959 Receptive fields of single neurones in the cat's striate cortex. In J Physiol. 1959 Oct;148(3):574-91. doi: 10.1113/jphysiol.1959.sp006308.
- Eleanor M. L. Scerri, et al., 2021, Continuity of the Middle Stone Age into the Holocene, In Scientific Reports volume 11, Article number: 70 (2021)
- Catherine Hobaiter and Richard W. Byrne, 2014, The Meanings of Chimpanzee Gestures. In Current Biology Volume 24, Issue 14, 21 July 2014, Pages 1596-1600
- Richard Moore, 2014, Ape Gestures: Interpreting Chimpanzee and Bonobo Minds. In Cell Press. Volume 24, Issue 14, 21 July 2014, Pages R645-R647

R.W.Bryne, et al., 2017, Great ape gestures: intentional communication with a rich set of innate signals. In *Anim Cogn* DOI 10.1007/s10071-017-1096-4

Jeff Hawkins et al., 2005, *On Intelligence*. Published by Times Books.

Del Signore, 2014, Measuring and Simulating Cellular Switching System IP Traffic. In *Bell Labs Tech. J.*, 18: 159–180. doi:10.1002/bltj.21651

<http://grammar.ccc.commnet.edu/grammar/adjectives.htm>

repository of code used to generate figures:

<https://github.com/kwd2/graph1>