Generation of Simple Sentences in English Using the Connectionist Model of Computation

Jugal Kalita and Lokendra Shastri
Department of Computer and Information Sciences
University of Pennsylvania, Philadelphia, PA 19104

March 13, 1987

Abstract

This paper discusses the design and implementation of a connectionist system for generation of well-formed English sentences of limited length and syntactic variability. The design employs several levels of interacting units for making appropriate decisions. It uses a simple technique for specifying assignment of input concepts to roles in a sentence and also has a *reusable* subnetwork for the expansion of noun phrases. The same NP-subnetwork is used for the expansion of noun phrases corresponding to the subject as well as the object phrases of the generated sentences. The input to the system consists of parallel activation of a cluster of nodes representing conceptual specification of the sentence whereas the output is in the form of sequential activation of nodes corresponding to the words constituting the sentence. The system can produce simple sentences in both active and passive voices, and in several tenses. Results of a simulation experiment performed are also included.

1 Introduction

It is generally accepted that text generation involves three distinct sequential phases – content determination, text planning and surface generation. Content determination involves identifying information that needs to be included in the generated text. Text planning is concerned with appropriately sequencing the intended contents in order to achieve coherence in the generated text. The final phase — *surface generation* produces the actual sentences given their internal representations. In this paper, we are not involved with the first two phases. Our emphasis is on the generation of well-formed sentences assuming that the initial two phases have been successfully performed.

Currently, there are several approaches to surface generation. These include the approach based on functional grammar as used by McKeown in her TEXT system [McKeown 85], the propositional logic based system employed by Appelt [Appelt 83] and the stepwise refinement approach taken by McDonald in the MUMBLE system [McDonald 83]. The functional grammar approach is non-deterministic; the generation of a sentence from a conceptual specification involves the process of unification which is slow and inefficient taking potentially exponential computation time. Appelt assumes homogeneity of various decision processes; his approach which requires implementation of theorem proving techniques is also not easily amenable to a parallelism. MUMBLE employs several levels of processing to achieve its goal. Consequently, processing requirements are relatively simple at each level enhancing efficiency and modifiability. It is deterministic in the nature of processing involved.

Our approach to surface language generation is similar to that of McDonald. We have modeled the generation process as a hierarchy of processing levels. Decisions have to be made at each of these levels until nodes corresponding to the words constituting the sentence are activated in an appropriate order. We attempt to implement the various processing levels by using the connectionist model of computation in the spirit of [Feldman et al 82]. The techniques employed in our implementation have been influenced by Cottrell's system [Cottrell 85] for word-sense disambiguation and parsing. We successfully generate English sentences of limited length and limited syntactic variability given a non-linguistic specification of their contents.

2 The Levels of Processing

As indicated earlier, there are several levels of computation in the generation of a well-formed sentence from its conceptual specification in our generational paradigm. In this respect, our chosen approach is similar to the approaches to perceptual processing reported in [McClelland et al 81,Sabbah 85] where each level of processing is concerned with forming representation of the input at a different level of abstraction. In our system, levels represent various decision steps that need to be taken in order to generate a sentence.

The main levels of processing employed in the system are

- input level
- realization-class level
- choice level
- constituent level
- morphology level

The relationships among the various levels are shown in figure 1.

Each level of processing is carried out by one or more specially designed cluster of nodes. The *input* level nodes represent a communicative goal (along with some constraints to be discussed later). This information is in the form of a non-linguistic conceptual specifications of what needs to be conveyed through an intended utterance. Examples of concept level nodes are *concept-eat*, *concept-monkey-1*, *concept-banana-1* etc.

Each concept node is linked to a *realization-class* node. Several concept nodes may be linked to the same realization-class node, but each concept node is connected to a unique realization-class node. The realization-class node which is connected to a concept node determines how that concept will be translated into English. The translation of all concepts whose corresponding nodes are connected to a realization-class node proceed in an identical fashion. Examples of realization-class nodes are *svo* and *individual-item*. The *svo* node requires concept nodes linked to it be translated into a sentence where the subject, verb and the object are all present. Concept units corresponding to transitive verbs such as *eat, give* etc., are connected to the *svo* node. Concept nodes such as *concept-monkey-1*, *concept-baboon-1*, *concept-banana-1*, which denote individual objects that can be expressed in English in terms of noun phrases, are each connected by an excitatory link to the *individual-item* node in the realization-class level.

Each realization-class node has activation links to one or more *choice* level nodes. The choice nodes connected to a realization-class node form a *winner-take-all* network. In other words, only one of them can be active at any instant of time. Each choice node connected to a realization-class

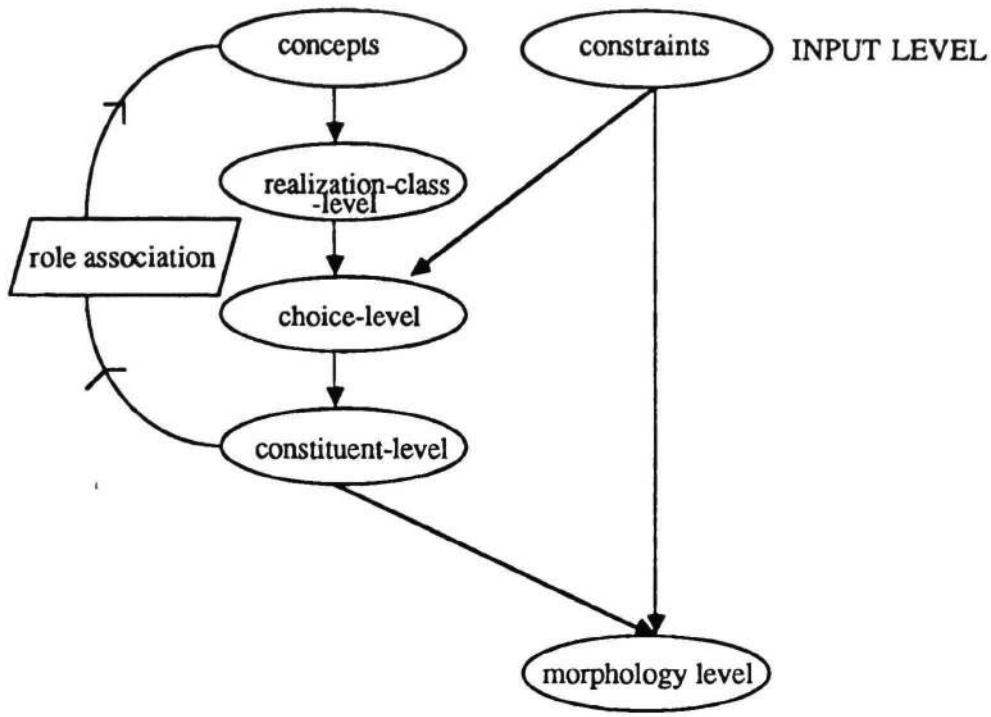

Figure 1: **Various levels and their interactions**

node represents a different grammatical way of realizing a concept linked to that realization-class node. For example, the realization-class node *svo* is connected to two choice nodes: *active-svo-choice* and *passive-svo-choice*. This allows a concept such as *concept-eat* (whose corresponding node is connected to the *svo* node) to be translated into either an active or a passive sentence after its subject and object roles are filled.

The choice nodes have other activation links from *constraint-specification* nodes (which are input nodes) incident upon them. Activation on each input link is necessary for a choice node to become active. The constraint-specification nodes must be activated by the text planner before the process of surface generation is started. Examples of constraint-specification nodes include *current-voice-is-active*, *current-aspect-is-perfect* etc. In order for the *active-svo-choice* node to be active, it must receive activation inputs both from the *svo* node (a realization-class node) and the *current-voice-is-active* node (a constraint-specification node).

A choice node has activation links to one or more *constituent* level nodes. The constituent nodes specify the roles that need to be filled in order to achieve a particular choice of realization of an input concept. Examples of constituent nodes connected to the choice node *active-svo-choice* are *subject-slot*, *active-svo-verb* and *object-slot* nodes. The activation of the constituent nodes is sequentialized. Thus, the translation of a concept node connected to the *svo* node (a realization-class node), assuming the *current-voice-is-active* node is active, involves filling in three roles in sequence: subject, verb and object. Sequencing is accomplished using *sequentialization* nodes discussed later.

Each constituent node is either connected to one or more nodes that handle morphology or a cluster of nodes that specify role association. Activation of a constituent node may result in excitation of one or more word nodes in a predetermined sequence. If a constituent node is connected to a node that specifies role assignment, then its activation starts expansion of another realization-specification node corresponding to the conceptual specification of a part of the sentence (e.g. the subject or object role of the sentence).

The morphology handling level contains two types of nodes: *generic-word* and *word* nodes. It

is assumed that activation of a word node leads to the word being spoken aloud or written out. A generic-word node represents a word whose lexicalization is dependent upon the grammatical context. For example, the node *generic-word-eat* is connected to the word nodes *word-eat* and *word-eats*. It should be noted that complex forms of verbs such as *has been eating* (i.e., three word nodes: *word-has, word-been* and *word-eating* whose activation is sequenced) are also connected to the *generic-word-eat* node. Depending on constraints such as current tense, current voice and current aspect, one of these sets of word nodes will be activated.

In addition to activation links from binder and constraint-specification nodes, the morphology nodes may have activation links from concept nodes via the cluster of nodes specifying role association. The realization-specification node to which a concept node is connected, determines the structure of the phrase or clause that results from the concept's translation through its links to choice and constituent nodes. One or more of the word nodes which need to be activated to fill in the slots in the translated phrase or clause may be determined directly by the concept node itself by having activation links onto specific generic-word or word nodes.

3 Implementation Details

Each of the levels of processing discussed in the previous section has one or more unit types associated with it. Thus, we have the following unit types: concept, constraint-specification, realization-class, choice, constituent, generic-word and word.

Input nodes which include concept and constraint-specification nodes are activated by the text planner. They are activated simultaneously and thereafter they drive the process of generation.

Generation of a well-formed sentence involves activating the nodes which correspond to words in the sentence. However, parallel activation of word nodes cannot be considered as generating a sentence. Sequentialization of activation of word nodes is imperative in order to generate a meaningful sentence. It should be noted at this point that a connectionist model is inherently parallel, and hence additional efforts are required to achieve the sequentialization needed by the nature of the problem under consideration. The sequencing technique discussed next is similar to the one employed in [Cottrell 85].

In order to achieve appropriate word sequencing, a new unit type has been defined: the *sequentialization* type. Sequentialization nodes also assist in meeting the crucial requirement that only one word node remain active at any instant of time. Feedback from a corresponding sequentialization node turns off a word node a few cycles after it has been activated. Sequentialization nodes also facilitate the deactivation of units which have already participated in the generation process and are no longer required.

Each unit in the network has several sites. The nature of these sites is dependent on the unit type. Among the sites are: *or*, *and*, *expansion-completed* and *reuse-site*. An *or* site computes its value to be the highest among all its weighted input signals, whereas the computation carried out in at the *and* sites involves selecting the minimum of its weighted inputs as its value. The function of the other two sites are discussed later.

The units can be in three possible states. They are *initial-inactivation*, *activation* and *final-inactivation*. All nodes are initially in the initial-inactivation state. A node is switched to the activation state on receiving appropriate excitation signals at its *or* or *and* sites. Once in the activation state, a unit remains so until such time when a signal is received at its *expansion-completed* site from a sequencer unit. This forces a unit to *final-inactivation* state. Thus, when all activity ceases, the nodes which participated in the processing are left in *final-inactivation* state.

The unit behavior discussed thus far is sufficient to generate simple sentences, but in order to

achieve better resource utilization, it is necessary to introduce another site called the *reuse-site*. A positive input at this site of a unit in *final-inactivation* state causes the unit to change its state to *initial-inactivation* state so that the unit can be reused. Reuse-sites are used only for the nodes in the subnetwork for noun phrase expansion. This enables us to use the same noun phrase expansion subnetwork for expanding the subject as well as the object of a sentence. Using two separate subnetworks for noun phrase expansion – one for the subject and the other for the object would lead to a wastage of resource. Since the task is essentially the same, we have decided to design a reusable NP-subnetwork. When a noun phrase expansion is complete, a specially designated unit in the NP expansion subnetwork gets activated. This unit, then, sends a positive activation to all units in the noun phrase subnetwork at the *reuse-site* as a result of which the units become ready to resume their activity.

This, in our view, is a significant feature of our implementation. Supposing we are generating an active sentence, the subject NP needs to be expanded first. In order to achieve this, the *subject-slot* constituent node is activated. This initiates the expansion of the subject NP. Once this expansion is complete, all nodes that participated in this expansion in the NP subnetwork are reset by a signal at the *reuse-site* as explained earlier and are available for reuse. After subject NP expansion, the active verb phrase is generated. Following this, the same NP expansion subnetwork is used to generate the noun phrase corresponding to the object.

In order to generate active as well as passive sentences, we have a *passive-svo* choice unit connected to the realization class unit *svo*. The passive choice unit gets activated only when it receives activation from the realization class unit *svo* and the constraint specification unit *current-voice-is-passive*. The subnetwork employed for generation of subject and object noun phrases for active sentences is also used for passive sentences without any modification. New sequencing nodes have been introduced so that the expansion of the object phrase precedes that of the verb and agent phrases for the passive case. Units for handling passive forms of verbs have also been implemented.

The distinction between the two inactive states is introduced to prevent oscillation of a node between active and inactive states. This requirement is extremely important, in particular, for word nodes in order to achieve proper sequentialization of utterance. In our implementation, it was required of a unit to remain in inactive state once there is a transition from active to inactive state to prevent unexpected potential and state oscillations. A node which is inactive to start with plays its appropriate role in the process of generation by becoming active on receiving excitatory inputs. Consequently, it activates other units, and in turn, it becomes inactive again (final-inactivation). However, the arrival of a positive input at the *reuse-site* causes a unit in final-inactivation state to make a transition to initial-inactivation state enabling its participation in repetitive processing.

The mechanism by which the input to the network is specified is described below with the aid of figure 2. We should be able to associate concepts with roles such as subject and object in a sentence. Corresponding to each object that can fill in the subject/object role of a sentence, we have a concept node such as *concept-monkey-1*, *concept-banana-1*, etc as discussed earlier. Each of these concept nodes is connected to the realization-class unit *individual-item*. Each of these concept units is also connected to two binder units for the purpose of role association. Consider the concept unit *concept-monkey-1*. It is connected to two units labeled *e-s-1* and *e-o-1* in the diagram. *e-s-1* is called a *subject-binder* unit; *e-o-1* is an *object-binder* unit. There are two such units for each concept that can fill the role of subject/object. In this example, *e-s-1* plays a role in associating *concept-monkey-1* unit to the subject role; and *e-o-1* helps associate *concept-monkey-1* to the object role. It should be noted that each *subject-binder* node (for each concept) has an activation link from the *subject-slot* unit (which is a constituent unit driven by the svo-active/svo-passive choice units for the realization-class unit svo). Similarly, the object-binder node for each concept has an activation link from the *object-slot* constituent unit.

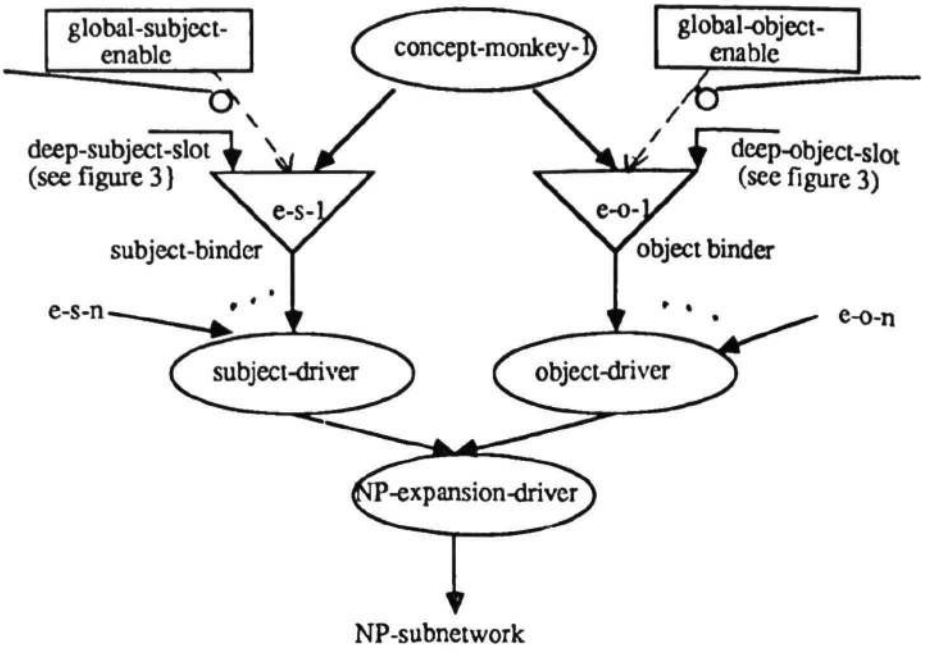

Figure 2: **Binding of Concepts to Roles in Input Specification**

Several other units are also used in this process. Two such units are the *global-subject-enable* and *global-object-enable* nodes. These nodes always kept active. There is a link from *global-subject-enable* node to the *subject-binder* node for each concept node. There is a similar link from *global-object-enable* to the *object-binder* node for that concept. Initially, each such link is in a *disabled* state. Appropriate links are enabled before processing a sentence to specify role assignments. Enabling of role assignment links is done at input time before processing is initiated. This is a required in order to correctly specify at a conceptual level which concept plays what role in the sentence to be generated. If, however, our system needs to be interfaced with higher level generation systems that perform text planning, such enabling of links will have to be performed automatically. This is a problem which future research must address.

In order to assign the *concept-monkey-1* unit to the subject role, the link from the *global-subject-enable* to the subject-binder unit for *concept-monkey-1* (here, unit *e-s-1*) is enabled at input time. (Please note that all other links from *global-subject-enable* unit to all other subject-binder nodes are still disabled.) And, similarly to assign another unit (say, *concept-banana-1*) to the object role, the link from *global-object-enable* unit to the object-binder unit for *concept-banana-1* is also enabled. So, the crucial step in specifying the input properly constitutes enabling the linkages appropriately before the processing of a sentence starts.

The subject-binder and the object-binder nodes are such that they need all their inputs to get activated. A subject-binder node provides excitatory input to the *subject-driver* node. Excitation from any subject-binder node excites the *subject-driver* node. Similarly, there is an *object-driver* node which receives excitatory input from the object-binder nodes. The *subject-driver* and the *object-driver* nodes have excitatory inputs to an *np-expansion-driver* node. This node drives a node called the *binding-completion-signaler* which initiates the expansion of the *choice* nodes connected to the *individual-item* realization-class node. Thus, excitation of the *binding-completion-signaler*

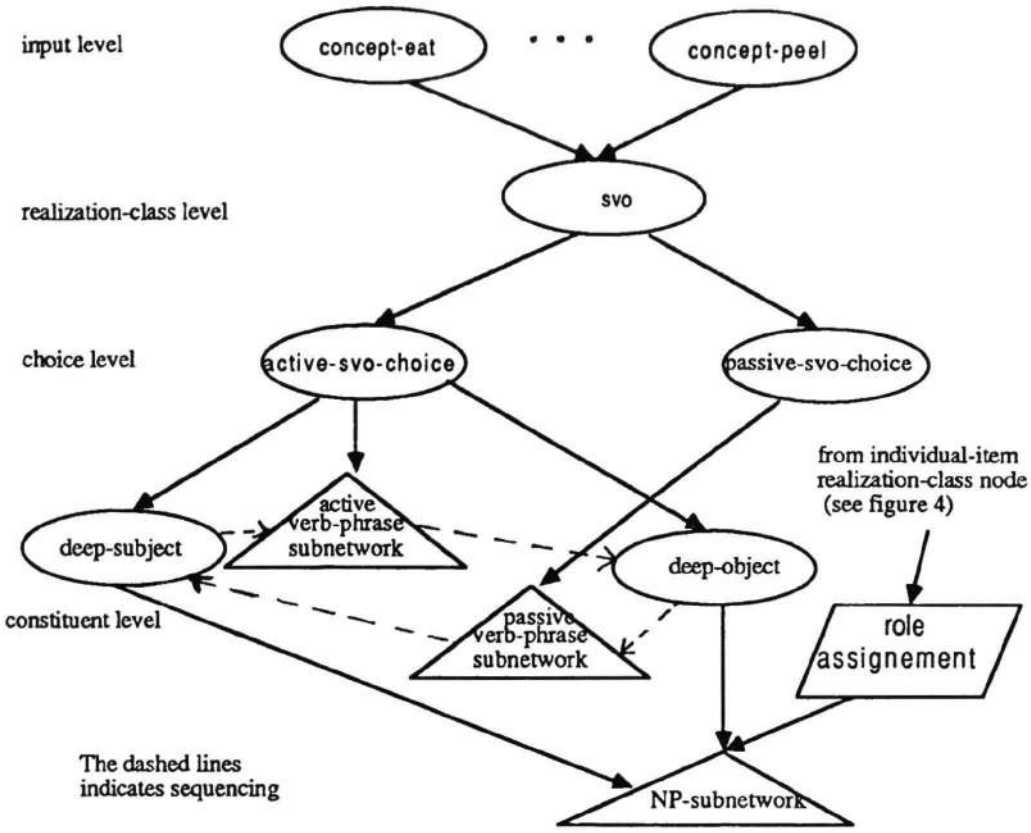

Figure 3: **A section of the relevant portion of the network** (also see figure 4)

unit which sits between the realization-class node *individual-item* and its choice units initiates the excitation of units that leads to the expansion of the noun phrase corresponding to the subject or object of a sentence.

The choice unit *active-svo* has excitatory inputs to *subject-slot, active-svo-verb* and *object-slot* units – the activation of these units is sequentialized. Thus, during the expansion of the subject, only the *subject-slot* unit is active; as a result, the subject-driver node is activated (object-driver node is off). This leads to the generation of a noun phrase corresponding to the subject of the sentence. Similarly, when the *object-slot* unit is active, the *object-binder* unit corresponding to the object concept gets activated. This activates the *object-driver* node which finally results in the generation of a noun phrase corresponding to the object concept.

4 An Example Simulation

We will now briefly run through an example simulation showing the various steps involved in generating a sentence given the conceptual specification of its contents. The network for the simulation was built using the ISCON simulator [Fanty 85]. The network has about one hundred units. The relevant portions of the network are shown in figures 3 and 4. Sequentialization nodes are not shown here.

Initially the input units corresponding to the conceptual level description of the sentence to be generated are activated. For example, in order to generate the sentence *A monkey is eating a banana*, we activate the units corresponding to *concept-eat*, *concept-monkey-1* and *concept-banana-1*. We also activate the constraint-specification units – *current-voice-is-active*, and *current-aspect-*

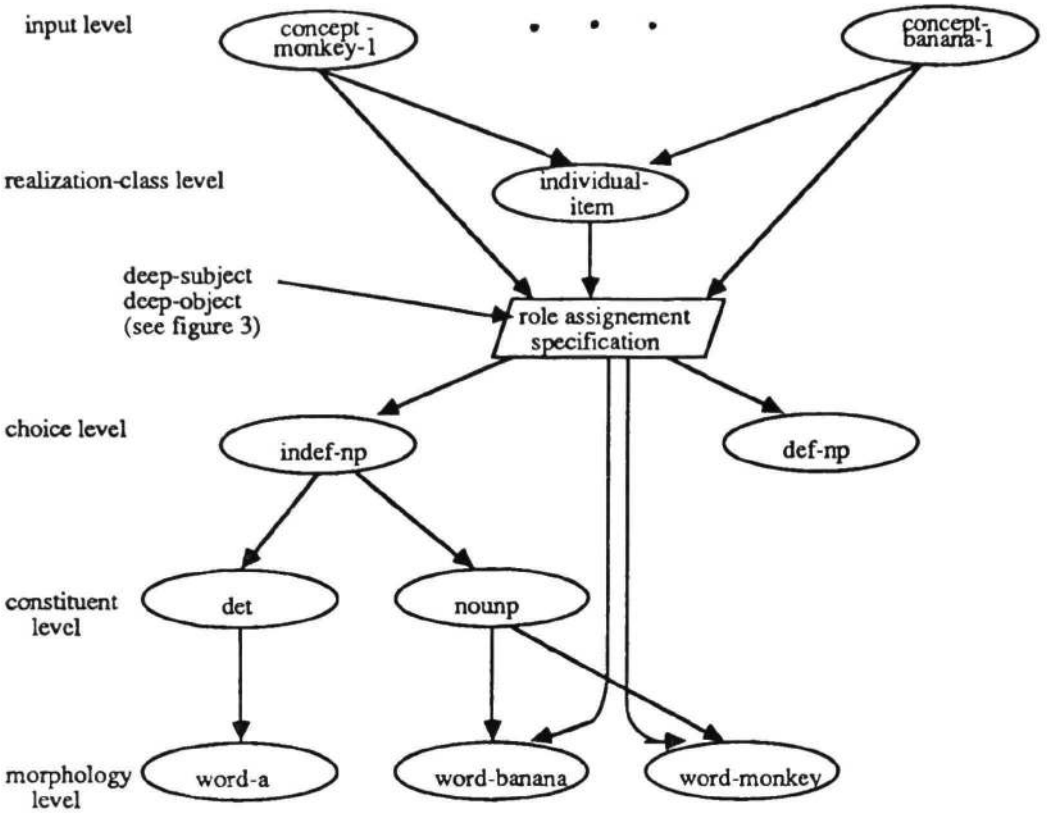

Figure 4: **Section of the network handling noun phrases**

is-continuous. The units *global-subject-enable* and *global-object-enable* are also activated and the links from *concept-monkey-1* to its subject-binder (viz., *e-s-1*) and from *concept-banana-1* to its object-binder (viz., *e-o-2*) are enabled.

Activation of the *concept-eat* unit activates the realization-class unit *svo*. This along with the fact that the *current-voice-is-active* unit is on, turns on the choice unit *active-svo-choice* which, in turn, sequentially activates the three constituent units connected to it, viz., *deep-subject*, *active-svo-verb* and *deep-object*.

Meanwhile, activation of the concept-level unit *concept-monkey-1* had turned on the realization-class unit *individual-item* which feeds an activation link to the *binding-completion-signaler* node which however, remains inactive. Among the constituent units, the *deep-subject* unit is activated first. Now, the set of units which participate in the binding process comes to play its role. Since *deep-subject* unit is active, the unit *e-s-1* receives activation from all its inputs, and this results in the binding of the *concept-monkey-1* unit to the subject role. Finally, as a result of binding, the *np-expansion-driver* unit provides activation to the *binding-completion-signaler* unit which becomes active. It feeds the activation link to the choice unit *indefinite-np* which is turned on. It provides excitatory input to the constituent units *determiner* and *nounp* which are activated resulting in the sequential activation of the units *word-a* and *word-monkey*. Expansion of the subject is now complete. A special sequencer node is activated; it resets the various nodes involved in the expansion of the subject making the NP subnetwork available for reuse for the object phrase expansion, and also starts the expansion of the verb phrase by sending an excitatory signal to the *active-svo-verb* constituent unit.

The expansion of the verb phrase of the sentence leads to the utterance of the word units *word-is*

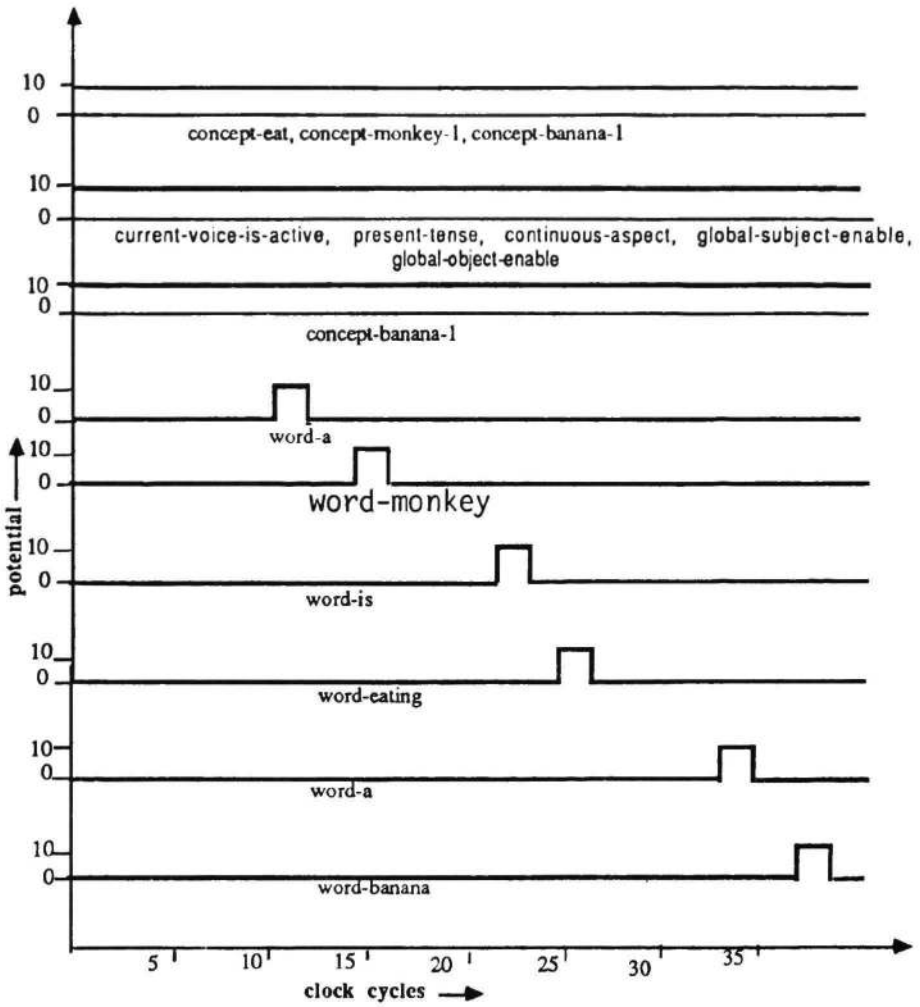

Figure 5: **Graph of potentials of relevant units**

and *word-eating* in sequence. All units involved in the expansion of the verb phrase are turned off, and expansion of the noun phrase corresponding to the object of the sentence is initiated resulting in activation of the word units *word-a* and *word-banana* in sequence. This completes generation of the whole sentence and is followed by deactivation of all units that took part in the process of generation and are still active. Finally, a specialized sequentialization unit is turned on. Activation of this unit can be used to generate an appropriate pause (in case of speech) or the punctuation symbol '.' in case of written text.

A graph showing the potentials of the concept, restriction-specification and the word units for this example are shown in figure 5. This clearly shows that the words that constitute the sentence are activated in proper sequence. Each word unit is active for a short period of time during which it is assumed to be spoken or written. Examples of other sentences that can be generated by our system include *A monkey has been eating a banana, A banana is being eaten by a monkey, A baboon was peeling a banana*, etc.

5 Discussions

The sentences generated by the current network are all of the form *svo* where a sentence contains a subject, a verb and an object. They can be in active or passive voice and in several different tenses. All noun phrases generated have a determiner followed by a noun. All verbs used so far are transitive. Also, all passive sentences currently generated have the *by*-phrase following the verb.

We need to incorporate intransitive verbs and allow the agent phrase to be optional in passive sentences. Future research is also necessary in many other fronts, such as, incorporation of relative clauses, repeated constituents such as prepositional phrases and adjectives, anaphoric pronominal phrases, etc.

An open problem that needs to be addressed is regarding mechanisms for interfacing our surface generation system with other systems that perform content determination and text planning in natural language generation. Our approach to specifying this interface is simple, but inefficient. A more sophisticated technique might involve using *exploded case frames* introduced in [Cottrell 85]. This will allow us to prevent the generation of such absurd sentences such as *A banana is eating a monkey* by imposing selectional constraints on the fillers of roles of the verbs. This will necessitate the incorporation of an elaborate knowledge base describing the concepts that are available in our domain of interest. The specification and organization of such a knowledge base such that responses to pertinent queries can be elicited efficiently can be modeled after [Shastri 85]. Additionally, in the current implementation, we need two binder units – a subject-binder and an object-binder for each concept that can fill the subject/object role of a sentence. Since, the number of such concepts is usually very large, it will be worthwhile to investigate other approaches to the problem of binding. In general, this problem is an instance of *the binding problem* in connectionist systems. This problem is not specific to our work, and any elegant solution to it will be applicable here.

Currently, we are working on automating the process of building up the network from a declarative specification of the realization-specifications, choices and constituents and the bindings. This will help in adding new sentence types and component structures easily for the purposes of testing and development. It will also involve automatically identifying units which can be shared among the various constituent schemata and how sequentialization nodes are to be interconnected.

Another direction in which we intend to pursue further research is regarding the choice of lexical tokens in the generated sentences. Lexical choice is governed by a large number of factors – syntactic, semantic and pragmatic. These include considerations such as relevant characteristics of the speaker and the hearer, their interpersonal goals, the conversational setting, etc. Identification of such factors, and a careful analysis of the nature of their interplay is essential in order to be able to develop a systematic theory of lexical choice. It is believed that different factors provide different levels of evidence for the selection of competing words, and these have to be appropriately combined in order to be able to select a particular word over its competitors.

References

[Appelt 83] Appelt, D.E., Planning Natural Language Utterances, *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, 1983, pp. 59-62.

[Cottrell 85] Cottrell, G.W., *A Connectionist Approach to Word Sense Disambiguation*, TR-154, Department of Computer Science, University of Rochester, 1985.

[Fanty 85] Fanty, M., *Connectionist Network Simulation Tools*, Technical Report, Department of Computer Science, University of Rochester, 1985.

[Feldman et al 82] Feldman, J.A., and Ballard D., Connectionist Models and their Properties, *Cognitive Science*, Volume 6, 1982, pp. 205-254.

[McClelland et al 81] McClelland, J.L., and Rumelhart, D.E., An Interactive Activation Model of Context Effects in Letter Perception: Part 1, An Account of Basic Findings,

Psychological Review, Volume 88, Number 5, September 1981, pp. 375-405.

[McDonald 83] McDonald, D.D., Description Directed Control: Its Implications for Natural Language Generation, *Computers and Mathematics with Applications*, Volume 9, No. 1, 1983, pp. 111-129.

[McKeown 85] McKeown, K., *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*, Cambridge University Press, 1985.

[Sabbah 85] Sabbah, D., Computing with Connections in Visual Recognition of Origami Objects, *Cognitive Science*, 9, 1985.

[Shastri 85] Shastri, L., *Evidential Reasoning in Semantic Networks: A Formal Theory and its Parallel Implementation*, Ph.D. Thesis, Department of Computer Science, University of Rochester, 1985.