Artificial Intelligence Journal, 57, pp 183-225, 1992.
near-final draft, lacking figures

A Parallel Approach to Syntax for Generation

Nigel Ward
University of Tokyo

Abstract

To produce good utterances from non-trivial inputs a natural language generator should consider many words in parallel, which raises the question of how to handle syntax in a parallel generator. If a generator is incremental and centered on the task of word choice, then the role of syntax is merely to help evaluate the appropriateness of words. One way to do this is to represent syntactic knowledge as an inventory of "syntactic constructions" and to have many constructions active in parallel at run-time. If this is done then the syntactic form of utterances can be emergent, resulting from synergy among constructions, and there is no need to build up or manipulate representations of syntactic structure. This approach is implemented in FIG, an incremental generator based on spreading activation, in which syntactic knowledge is represented in the same network as world knowledge and lexical knowledge.

1 Introduction

This paper explains how, in the process of trying to build a good natural language generator, I discovered that neither explicit choice of syntactic constructions nor assembly of representations of syntactic structures is necessary.

This section briefly characterizes the generation task, explains why it requires parallel processing, discusses the implications and requirements for a syntactic mechanism, and sketches out a proposal. Subsequent sections present a simple generation program (2), explain the details of its treatment of syntax (3, 4, 5), compare this to other approaches to syntax for generation (6, 7, 8), and discuss the significance of the results (9, 10).

1.1 Two Aspects of the Generation Task

The generation task is uncontroversially a process of getting from thoughts, goals, or meanings to words. Traditionally this has been seen as a process of mapping input to output, but this metaphor obscures some important issues. The general problem is that thoughts (or whatever the input is) do not always map easily or trivially onto words. This is not obvious to speakers, since people are very good at producing language, nor to hearers, since people automatically filter out disfluencies in speech. However there are occasions where the difficulty of producing language is clear.

When revising a document it is not uncommon for the desire to change one word to necessitate the rewriting of a sentence or an entire paragraph. This illustrates the problem of dependencies among choices. That is, in order to determine how to express one part of the input a generator must consider the way the surrounding utterance will turn out. While the purpose of generation is clearly to express thoughts, the language at hand strongly constrains how this is done. A concrete, if simple, example of this is collocations, such as *high winds* (versus *swift currents*), where the choice of the adjective depends on the choice of the noun.

Another occasion where generation is transparently difficult is when helping a non-native speaker to produce good English. To decide what English word to use, *the* vs *a*, for example, it is sometimes necessary to spend several minutes clarify exactly what he wants to say or the nature of the thing he is describing. That is, arbitrarily obscure factors can affect word choice. Another occasion where the difficulty of generation becomes apparent is when trying to evoke complex images or feelings. For example, a novelist sketching out a scene will mention a few salient aspects, leaving the reader to apply his own knowledge of graveyards or train stations or whatever, and come up with his own image of the scene. Similarly if, in the course of translating, the translator "goes behind" the text to a richer interpretation, he then faces the task of producing a translation where only certain aspects of this can be included explicitly.

In these cases it is clear that the "input" to the generation process is much richer in information than the resulting output. That is, there is typically much more information present and potentially relevant than is explicitly expressed in the final utterance. This has not yet arisen as a practical issue since current AI programs which use generators do not have complicated things to say. Yet this situation will change; we can look forward to the day when generators will be faced with tasks like explaining the rationale behind advice from an expert system with compex causal and background knowledge, or expressing concisely the information that a good parser/understander of, say, French, should be able to extract and infer from an input sentence.

1.2 The Need for Parallelism

This paper discusses five kinds of parallelism for generation: part-wise parallelism, competitive parallelism, knowledge-source parallelism, evaluative parallelism, and synergistic parallelism. This subsection explains how the issues discussed above suggest the use of the first four kinds of parallelism; the last kind is discussed in §6.1.

In order to respect dependencies among choices of words, a generator must consider words for various parts of the output in parallel. Such "part-wise parallelism" is also the natural way to generate from inputs rich in information, since the components of the input can all be "goals" active in parallel, each suggesting words and constructions appropriate for it. Use of part-wise parallelism contrasts with compositional generation, as implemented, for example, with serial traversal of the input.

In case the "first choice" for the realization of something turns out inappropriate in the larger context, due to dependencies, it is necessary to have alternatives available. In the earlier example, to express the velocity of motion, the generator needs access to *strong*, *high*, *fast*, *swift*, and so on. Thus there is a need for parallel consideration of alternative, competing, words. Using such "competitive parallelism" contrasts with considering only one candidate output word at a time, and considering other alternatives, perhaps by backtracking, only if necessary.

Given that the choice of a word can be affected both by considerations of the meaning to express and by considerations of compatibility with other choices, it seems reasonable to have both types of information available simultaneously while choosing words. Such "knowledge-source parallelism" contrasts with applying different kinds of knowledge in different stages.

Part-wise parallelism and competitive parallelism, taken together, require the generator to consider a great many words in parallel. In the end, of course, the generator must choose a relatively small set of words to emit. To do this it must somehow evaluate each candidate word. That is, the generator should come up with an estimate of

the appropriateness of the given word, based on all the relevant factors of the input and all the other words under consideration. The most obvious way to do this is to evaluate all the evidence in parallel. Such "evaluative parallelism" presupposes knowledge-source parallelism; in order to use all the knowledge in parallel, it must all be available in parallel. Most generators instead filter out candidate words in stages, for example, by first considering all semantic factors, and then considering collocational properties and so on.

To come up with a score for the appropriateness of a word it is simplest to reduce all considerations to a common currency. That is, in order to make commensurate all the diverse factors relevant to a word, it seems simplest to reduce all considerations to numbers, and use some numeric scheme for the combination of evidence. (This is important also for dealing with contradictory factors. If the task of generation is stated as one of simultaneously meeting the two goals "express the input" and "produce a good sentence", it is clear that these will sometimes conflict. For example, the goals of clarity and conciseness are often in tension. To resolve such conflicts the relative strength of different considerations must be quantified.)

So far, then, my image of the generation process is that there is a very complex, rich input, and all potentially relevant words get considered and scored for appropriateness, with respect to the input and to the other words that are likely to be chosen. (Indirect support for the validity of this analysis is that introspection, pauses, and speech errors suggest that human speakers also exploit the kinds of parallelism discussed above (Ward 1989a).) To produce the final output the generator must select some (relatively small) mutually compatible subset of the most highly activated words. This process of course involves syntax, which will be discussed shortly.

1.3 Traditional Approaches

That the generation task requires parallelism has not been generally recognized. A historical reason for this might be the pervasive focus on structure rather than process. It is common to start work from a notion of the inputs that a generator must deal with, or from a (typically structuralist) theory of the outputs that a generator must produce. Most generation research work has grown out of one of these two tendencies; that is, most generation research is motivated by a theory of meaning representation or a theory of syntax. I have focused instead on the generation process itself.

At the practical level this neglect of parallelism is possible because the tasks currently handled by generators are not very demanding and because their knowledge is limited, as the remainder of this subsection explains.

One reason that current generation systems perform just fine without parallelism is that they have very limited target-language knowledge. That is, the vocabulary is tailored to the specific task (McDonald 1987), and tailored so that problems of dependencies among choices do not arise. For example, if the only words a generator has to refer to motion are *come* and *go*, then its task is not too challenging. If, however, it also knows about *approach*, *toward*, *to*, *float*, *along*, *movement*, *downstream*, and *bobbing*, all of which have different constraints on the neighboring words they can appear with, then the task is harder. In general, the more options a generator has, the more difficult the task of finding a compatible set of words.

The second reason that most generators do well without parallelism is the simplicity of the inputs. In many cases the input is essentially no more than a "disguised" version of the desired output; to produce that sentence involves mapping each concept onto a word or syntactic structure and each relation onto a syntactic relation, with a little bit

of tidying up for the sake of inflection and word order. (This of course also means that a generator does not need to have many linguistic options, so the problem of dependencies does not arise either.) While such simple mappings are fine for current applications, they will not be adequate for inputs richer in information.

Some will argue that "someday generators will have to deal with non-trivial inputs, but that needn't concern us now, since we can always slap on an extra module to pre-process the input into the format our generators expect". While hardly falsifiable as a claim, as a research strategy this is rather suspect. For one thing, divisions of the generation process (typically into more language-dependent and less language-dependent aspects) appear not to work, because linguistic and conceptual choices are all interdependent (Danlos 1984). That is, it appears that a pre-processor cannot sensibly decide "what to say" if it cannot access knowledge about the target language; nor can the generator proper make good choices of words and syntactic form if it only has access to a pruned-out version of the input. A second problem with this "module-by-module" research strategy is that it allows important problems to fall in the cracks between the modules and not get addressed at all. For the rest of this paper I will assume that the entire generation process, from thought to words, can and should be handled as one process.

Others will argue, in a similar vein, "well of course generators should be parallel, and when prices come down I'll get a technician to port my generator to run on parallel hardware". Most generators, however, rely rather heavily on a careful ordering of choices, and it is not clear that they can take advantage of parallelism. Moreover, since the parallelism is needed at the task level, not just at the programming language or hardware level, it seems wisest to use parallelism directly in the model of generation.

To summarize, as we expect more from generators, both in terms of what they can handle as inputs and the range of outputs they can exhibit, parallelism becomes more important. Analysis of the shortcomings and limitations of traditional approaches to generation, and how to surmount them, leads to a similar conclusion (Ward to appear).

1.4 The Role of Syntax, and a Proposal

So, it seems that a generator ought to consider many words in parallel. To produce the final output it should take the "best" set; this is where syntax enters the picture.

In fact, the problem can be simplified to that of choosing the best word for each point in time, and iterating. That is, generation can be incremental. This requires that one of the factors in the score of a word be its appropriateness for the current time. If this factor is weighted strongly enough so that each successive word choice is appropriate in context, then the result of a sequence of word choices will be an appropriate utterance. (There is a (superficial) contradiction between choosing a consistent set of words and choosing words one-by-one, but the contradiction only holds if the generator has absolutely no idea of what it will subsequently produce, that is, if it has no part-wise parallelism.)

Syntactic considerations must affect the appropriateness scores of words. Applying the principle of knowledge-source parallelism to syntactic knowledge, it seems reasonable to have syntactic knowledge active at the same time as other kinds of knowledge, and, applying the principle of evaluative parallelism, it seems best to have syntactic considerations be simply one more factor in a word's score. Thus a high-scoring word will typically be one appropriate according to all types of considerations, syntactic, semantic, and collocational.

Thus there is a need for a syntactic mechanism that can contribute to the evaluation of many words at once. No traditional model of syntax is suited for this. The usual way for a generator to apply syntactic knowledge is to choose among syntactic alternatives and

to build up syntactic structures. In such approaches the knowledge is not mobilized in a way that supports the evaluation of many candidate words, at least not for words whose choice could constrain the embedding syntactic form.

The key to the solution is to treat syntactic knowledge as an inventory of syntactic constructions, much like the inventory of words which constitutes the lexicon, as proposed in Construction Grammar (Fillmore 1988; Fillmore 1989). For generation, then, constructions and words are all linguistic resources, of which the speaker must choose a set which is appropriate for the input and which is mutually compatible.

This analogy between syntactic constructions and words, taken together with the model sketched out above, has several implications. Just as many words can be considered in parallel, so can be many syntactic constructions, both for competitive and part-wise parallelism. Just as the score of a word depends on how appropriate it is for the meaning to be expressed, so too for constructions. Just as the score of a word depends on its compatibility with other words under consideration, so does the score of a syntactic construction. Just as all the factors contributing to the score of a word can be evaluated in parallel, so can the factors contributing to the score of a construction. (Constructions are not, however, entirely like words. For one thing, they do not have phonemes attached and so need not be explicitly chosen; for another, constructions can affect the order of words.)

The rest of this paper fleshes out these ideas and demonstrates that they are workable.

2 A Simple Generator

Before explaining in detail how this approach to syntax can work, it is necessary to describe the context in which it has been implemented.

2.1 Basics

FIG, for "Flexible Incremental Generator", is a system based on spreading activation; that is, the computational mechanism is flow of "activation" in an associative network. Network structures encode lexical, syntactic, and world knowledge. Concepts, words, and constructions are represented as nodes, and relations among them are represented as links. Each node is a simple computational unit. The nodes have various amounts of "activation", which change over time. The computation in the system is done by the individual nodes in parallel: based on the amounts of activation it receives, each node computes its own next activation level and the amount of activation to transmit to its neighbors.

The general direction of activation flow in FIG is from concepts to words, with nodes representing syntactic constructions "redistributing" activation to words which are syntactically appropriate for the current situation.

The FIG algorithm is:

1. each node of the input conceptualization is a source of activation
2. activation flows through the network
3. when the network settles, the most highly activated word is selected and emitted
4. activation levels are updated to represent the new current state
5. steps 2 through 4 repeat until all of the input has been conveyed

*** Insert Figure 1 Hereabouts ***

An utterance is simply the result of the successive word choices made in each iteration of the loop; thus FIG is incremental in a strong sense. The five steps are summarized in

Figure 1.

Of course, this description of the control flow is not very informative — what actually happens in each step depends on the structure of the network, the syntactic aspects of which are explained below.

2.2 Characteristics of the Model

To indicate briefly how this model addresses the issues raised in §1: First, FIG can handle inputs which are arbitrarily complex and information-rich, since any number of nodes can be activated in the initial state. There is no need for special processing to deal with such cases. Second, thanks to links among nodes representing compatible choices, FIG's network tends to settle into a pattern of activation which causes a consistent set of choices to emerge. There is no need to explicitly reason about interactions and dependencies.

To mention some other nice properties of this model: To adapt a generator to work incrementally usually complicates it, FIG is simple precisely because it exploits the temporal nature of generation. Similarly, having more knowledge of lexical and syntactic options usually complicates a generator, but in FIG having more options does not mean more complexity, and if running on sufficiently parallel hardware would not even invoke a time penalty.

FIG is built as a "structured connectionist" (Feldman *et al.* 1988) (or "localist connectionist" or spreading activation) system for four main reasons. First, it suits the task; since parallel computation and the numeric combination of evidence are required. Second, a spreading activation model can be very simple; the flow of information can be modeled directly as the flow of activation, and there are no data structures, buffers, modules, or complex algorithm to design and justify. Third, it allows inspiration from several helpful metaphors: parallel intersection search by flow of activation through world knowledge, exploited in FIG for lexical access (Ward 1988); finding an optimum in the face of multiple constraints by settling, exploited in FIG for arriving a set of words and constructions that satisfies interdependencies; and soft match to prototypes, exploited in FIG for the handing of relational information (Ward 1992a). Fourth, it is possible to build structured connectionist systems by hand, which is not true of distributed connectionist (PDP) models, which must be trained; this is important because the state of the art in learning is such that, in order to make a system capable of learning a complex task like generation, it is necessary to make many simplifying assumptions, as can be seen in Miikkulainen's PDP model, where generation is little more than a mapping process (Miikkulainen & Dyer 1991). FIG exploits these advantages of connectionism only when they are truly advantages; it includes symbolic processes for subtasks where the use of spreading activation would bring no immediate advantage.

2.3 More About FIG's Task

FIG's inputs come from a simple parser of Japanese or are assembled by hand. From these representations FIG produces English and Japanese sentences. These inputs used are a far cry from the "information-rich inputs" which are really needed for the generation task (§1.1). However for lack of a source of better inputs, I have had to make do with simple ones, taking care to ensure that FIG does not unfairly exploit this simplicity so that it should scale up (§7.4).

FIG's outputs are all single-sentence "utterances", printed on the screen, intended to convey static meanings to a generalized, non-situated hearer or reader. Thus it does not

address other issues in generation, such as crafting text for a specific hearer with specific knowledge and goals, producing texts longer than a sentence, rewriting, producing prosody, and integrating language production with other means of communication, such as gesture.

2.4 Notation Conventions

The names of nodes which represent "notions" (concepts) are subscripted with "n", syntactic constructions with "c", words with "w", and relations with "r". Constituents end in "-n", where n is a number, "collector nodes" end in "-a" or "-ness", and nodes with neither subscript nor suffix represent syntactic categories. The names of nodes and links are set in bold when they appear in the text.

***Insert Figure 2 Hereabouts ***

Graphs are used to represent portions of FIG's knowledge network, for example Figure 2 shows the link between a node representing a notion and one representing a word. The numbers on the links are their weights.

Notions in FIG are usually given Japanese names. This may help remind the reader that the starting point for generation need not be something which is easy to map to the target language. The original reason is that the parser FIG works with is rather lazy (in a certain way appropriate for machine translation (Ward 1989b)). Node names are of course irrelevant to the operation of the program.

3 Basics of Syntax

To recapitulate the requirements on syntax, since FIG is centered on the task of word choice, syntactic knowledge must mediate the computation of appropriateness of words. In fact, this is its only role: there is no explicit syntactic choice, nor any building of syntactic structure. This section explains how syntactic knowledge is represented in the network and how, at run-time, these network structures affect word choice.

By way of overview: the basic idea is that, like other nodes, nodes representing syntactic constructions have activation levels which represent their current relevance. The activation sent from these nodes to various other nodes affects the syntactic form of the utterance. Each construction is represented as a set of nodes; the node for the construction as a whole is linked to nodes for constituents. These links have the special property that their weights change over time; in particular, the link to the currently relevant constituent has weight 1 and the links to the other constituents have weight 0. There is an update process which ensures that these weights reflect the current syntactic state as the generation of an utterance proceeds.

3.1 Constructions in the Network

A construction is a pairing of meaning and form. As a simple example, consider the Presentational Existential There Construction (Lakoff 1987), as in *there was a poor cobbler ...*. To simplify, this construction is used to introduce a new item into the discourse, and its constituents are the word *there*, a verb, and then a description of the item being introduced, in that order. Other examples of constructions are the Subject-Predicate Construction, the State-Change Construction, representing the intransitive valence of state-change verbs such as *break* and *die*, and the construction for *kick the bucket*. Following Construction Grammar (Fillmore 1988), FIG represents all syntactic knowledge using constructions.

***Insert Figure 3 Hereabouts ***

In FIG constructions and constituents are simply nodes. For example, knowledge about the Presentational Existential There Construction, is encoded as shown in Figure 3. There is a link from the concept **introductory**$_n$ to **pr-ex-there**$_c$, the node representing the construction. In turn, there are links from **pr-ex-there**$_c$ to its three constituents; these are ordered, although this is not shown explicitly in the diagram. From constituents there are links to nodes specifying their "content": **et-1** is linked to **there**$_w$ and **et-2** is linked to **verb**.

***Insert Figure 4 Hereabouts ***

As another example of a construction, Figure 4 shows FIG's definition of **purpose**$_c$, representing purpose clauses with *to*. This construction has two constituents: **pc-1** and **pc-2**, which are to be realized as the word *to* and a verb phrase with a bare verb, respectively.

***Insert Figure 5 Hereabouts ***

As another example, Figure 5 shows FIG's representation of the Subject-Predicate Construction. The first constituent of this construction is linked to semantic relations which specify the participatory properties that a concept should ideally have in order to be expressed in subject position. The details are explained in (Ward 1992a).

*** Insert Table 1 hereabouts ***

Table 1 lists some other constructions present in FIG. The syntactic knowledge presented here and elsewhere is intended simply to illustrate the representational tools and computational mechanisms available in FIG; I do not claim it to be more than a coarse approximation to the facts of English, nor necessarily the best way to describe those facts in a grammar.

3.2 Incoming Activation

This subsection discusses the factors which make a construction relevant, that is, the nodes which impart activation to constructions. There is nothing surprising about the sources of activation to constructions: to preview briefly, constructions are associated with their meanings, with words, and with other constructions.

There are links to constructions from concepts. For example, as seen above, the concept **introductory**$_n$ has a link to **pr-ex-there**$_c$. Activation can reach constructions via multi-link paths involving nodes representing world knowledge. For example, if **kuru**$_n$ (coming) is active, activation will flow via its "supertype", **motion**$_n$, to **dir-particle**$_c$, the construction which allows particles of direction, as in *come down* or *go away* or *roll back*.

There are also links from "relations" to constructions, for example, the **purpose**$_r$ relation is linked (indirectly) to **purpose**$_c$. Thanks to this link, whenever a node which fills a **purpose**$_r$ relation is activated, **purpose**$_c$ will become activated.

There are links from nodes representing genre and register to constructions, for example, there is a link from **slang**$_n$ to **kick-bucket**$_c$; this causes FIG to produce *Mary kicked the bucket*, instead of *Mary died*, when the node **slang**$_n$ is activated.

There are links from syntactic categories to associated "embedding" constructions; for example, there is a link from **cnoun-ness** to **determinat**$_c$, which represents the Determination Construction, responsible for the positioning of determiners before common nouns, as shown in Figure 6.

***Insert Figure 6 Hereabouts ***

Constructions are also associated with other constructions. For example, there is a link from the first constituent of **determinat**$_c$, the Determination Construction, to

genitive$_c$, the Genitive Construction, as shown in Figure 6. This link encodes the fact that the determiner can be a possessive noun, as in *John's peach*.

Figure 6 also illustrates that a construction can receive activation from several sources, for example **genitive**$_c$ receives activation from both syntactic and semantic considerations. All these sources of activation are considered in parallel and summed, that is, there is evaluative parallelism. Most generators, in contrast, consider one kind of factor for any given construction, for example, one construction may be chosen top-down (due to another construction) whereas another construction may be chosen bottom-up (due to its head).

3.3 Outgoing Activation

Constructions have their effects by emitting activation which indirectly reaches words. This subsection enumerates the ways in which this happens.

Many constituents activate syntactic categories. One use of this is to ensure that "syntactically necessary" words appear. For example, every count noun must be preceded by a determiner (in FIG's subset of English). Suppose that **peach**$_w$ is activated, perhaps because a **momo**$_n$ (peach) concept is in the input. **a**$_w$ and **the**$_w$ will then become activated thanks to the path from **peach**$_w$ via **cnoun-ness**, **determinat**$_c$, **dt-1**, and **article**, as seen in Figure 6. In this way the activation of a count noun causes an increase in the activation levels of articles. Provided that other activation levels are appropriate, this will cause some article to become the most highly activated word, and thus be selected and emitted. Note that FIG does not first choose to say a noun, then decide to say an article; rather the "decision" to use an article and to output it before the noun results simply from activation flow; to be specific, from the fact that **determinat**$_c$ causes articles to become highly activated at the right time.

As mentioned above, the most highly activated word typically becomes that way because it receives activation from both semantic and syntactic sources. Strong syntactic activation can compensate for relatively weak semantic activation. An obvious example of this is articles, which have little semantic motivation, but compensatingly high link weights on the path which brings them syntactic activation. Another example is verbal particles of direction, as in *she went down to the stream*. Here the direction can be expressed with one little syllable; since it can thus be expressed at little "cost", this information is relatively likely to appear even if not terribly important (in languages without this option, such information could perhaps only be expressed in a more complex structure, and hence might not be included in the output unless important to say). Accordingly, the weights of FIG's **motion-verb**$_c$ construction are such that verbal particles of direction get a great deal of syntactic activation. In this way constructions can affect which information is expressed explicitly.

Another reason that constituents are linked to syntactic categories is so that constructions can affect the syntactic form in which information appears. These can determine what part of speech to use, leading, for example to *criticism* instead of *criticize* as the subject of a sentence.

Constituents also have links to nodes affecting inflection. For example, the second constituent of **purpose**$_c$ is linked to the node **bare**$_n$, which causes the verb to appear in bare form. Constituents can also, of course, have links to closed-class words. For example, the second constituent of the Periphrastic Causative Construction is linked to the word *make*.

Constituents can also refer to concepts and relations. This is how FIG ensures that concepts appear in the right syntactic positions. For example, the first constituent of

genitive$_c$ is **possessor**$_r$, meaning that it passes activation to whatever nodes of the input are tagged as filling the **possessor**$_r$ relation. A more complex example is that the second constituent of **transitive**$_c$, representing the Transitive Construction, is linked to **affected**$_r$, and, with negatively weighted links, to **part-cause**$_r$, **responsible**$_r$, and **volitional**$_r$, which causes patients to be expressed as direct objects. Specifying constituents in term of concepts and relations is not common in linguistics, but such "semantic specification of constituents" has been shown useful for writing concise grammars (Jurafsky 1990).

A constituent may be linked to nodes which specify both form and content. For example it may activate both a syntactic category node and a relation node. To produce *float down*, the second constituent of **dir-particle**$_c$, the Verbal Particle of Direction Construction, must specify that "the direction of the going" be expressed as a "verbal particle". Thus this constituent is linked to **direction**$_r$ and **vparticle**. Activation will thus flow to an appropriate word node, such as **down**$_w$, both via the concept filling the **direction**$_r$ relation and via the syntactic category **vparticle**. Thus FIG tends to select and emit an appropriate word in an appropriate form. More commonly, however, a constituent is linked to only one node, and either the form or the content is determined by synergy with other constructions, as discussed in §6.1.

3.4 Word Order

Generating incrementally is the key to handling word order parsimoniously. If each successive word choice is appropriate in context, then the result of a sequence of word choices will be an appropriate utterance. Thus the activation level of a word must reflect not only its relevance in general, but also its current relevance at time t, for each moment t of the process of producing an utterance. In particular, words which are currently syntactically appropriate must be strongly activated. This also is the responsibility of constructions.

In FIG the representation of the current syntactic state is distributed across the constructions. To be specific, FIG represents the current relevance of syntactic constructions with activation levels, and their current status with "cursors". Each construction has a cursor which points to the currently relevant constituent. The cursor "gates" the links to constituents, so that the link from the construction to the current constituent has weight 1 and the links to all other constituents have weight 0. As a result, appropriate words become activated at appropriate times. For example, when the cursor of **determinat**$_c$ points to **dt-1**, articles receive a relatively large amount of activation. Thus, an article is likely to be the most highly activated word and therefore selected and emitted. After an article is emitted the cursor advances to **dt-2**, and so on.

In some cases there is a single construction which is responsible for specifying the ordering, as for determiners and nouns. In other cases the order is determined by the interplay of several constructions. For example, the fact that determiners precede adjectives is handled primarily by having the links from **cnoun-ness** to **determinat**$_c$ and to **adj-mod**$_c$ have weights of .71 and .7, respectively, which allows **determinat**$_c$ to activate articles more than **adj-mod**$_c$ activates adjectives. In general there is no central process which plans or manipulates word order; each construction simply operates independently. More highly activated constructions send out more activation, and so have a greater effect. But the output is always determined only by the simple rule "select and emit the most highly activated word", and so word order is emergent.

(In FIG all constructions are ordered — indeed, the presence of a cursor is the only thing which distinguishes constructions from other nodes. This does not mean that FIG

cannot handle "free word-order languages"; it simply means that constructions play less of a role when generating them.)

After a word is selected and emitted, two things must be done to ensure that the current syntactic state of affairs is accurately represented:

First, the cursors of constructions must advance as constituents are completed. Any number of cursors may advance after a word is output; for example, emitting a noun may cause both the **prep-phr**$_c$ construction and the **determinat**$_c$ construction to update. (The dotted lines in the figures show the relations between constituents and the nodes which signal their completion.) After all the constituents of a construction have been exhausted the construction "resets", and the cursor returns to its original position. This takes care of utterances with multiple "copies" for example, several noun phrases. (This does not handle center-embedded copies of constructions; I am assuming it is possible to write grammars which do not involve recursion.) Some constructions need no resetting, since they have only one constituent. For example **adj-mod**$_c$ siphons activation over to adjectives whenever a common noun is activated. Thus any number of adjectives can be produced, and all will appear before the noun, as in *the strong big boy* (other knowledge will be required to get the ordering right (§9)).

Second, constructions which are "guiding" the output should be scored as more relevant. Therefore the update process adds activation to those constructions whose cursors have changed. Because of this FIG, even though it does not make any syntactic plans, tends to form a grammatical continuation of whatever it has already output. An example of the importance of this activation is in the production of passive sentences. If the noun first output happens to express a concept which is tagged as affected and neither volitional, active, nor responsible, then the cursor of **passive**$_c$ will advance, **passive**$_c$ will become highly activated, and it will then activate the word *is* and then the inflection category past-participle, in order to form a passive sentence. If the patient is not expressed before the verb then **passive**$_c$ will not become highly activated and will send out insignificant amounts of activation.

4 Two Details

Since FIG has "part-wise parallelism", words and constructions for all parts of the output are active simultaneously. This means that there is the potential for undesired interaction between various sources of activation. This section discusses two potential problems and uses them to motivate some details of FIG's treatment of syntax.

4.1 The Locality Principle

So far I have explained how FIG chooses words that are syntactically appropriate and also appropriate for the input. There are outputs which meet both of these criteria but which are nonetheless inappropriate, for example *a large mountain and a small peach* when the input calls for *a small mountain and a large peach*, and, similarly, *the big boy went to the mountain* when the input calls for *the boy went to the big mountain.* Clearly the missing third criterion is that each word be appropriate for neighboring words (or, equivalently, concepts); this criterion is necessary to rule out such "crosstalk".

In FIG this criterion is met because related concepts always become highly activated together. This "locality principle" holds by virtue of the flow of activation across links among the nodes of the input. In the example above, **ookii**$_n$ (large) thus becomes activated

together with **momo**$_n$ (peach), not together with **yama**$_n$ (mountain). Because of the locality principle, FIG exhibits a kind of "focus of attention", that is, groups of related nodes become activated in succession, similar to the "focus of consciousness" of Chafe (1980). FIG's notion of focus differs from some notions of focus in that in FIG being in focus is not all-or-nothing, rather, a node is in focus to a degree.

The locality principle is important not just for word order but also for word choice. The choice of a word can turn on subtle features of context. For example, a destination relationship is expressed with *in* as the default but with *into* if the destination is a region. Here also there is potentially a problem of crosstalk: if a **region**$_n$ node is present anywhere in the input, it could cause the choice of *into* over *to*, regardless of whether that region is the destination of the motion in question. Thanks to the locality principle, **region**$_n$ will become strongly active and relevant when, and only when, related concepts, such as the relevant motion, are active.

4.2 Various Evidence-Combining Functions

FIG's basic rule is that the activation level of a node depends on the sum of the amounts of activation it receives for its neighbors. Some types of nodes combine evidence differently. One such type is nodes representing words; their activation levels depend on the product of the syntactic activation received and semantic activation received; this strengthens the tendency for the words selected to be both syntactically and semantically appropriate. ("Syntactic activation" means activation whose source is a syntactic construction and "semantic activation" means activation received more directly from the nodes of the input.) The rest of this subsection motivates and explains another special evidence-combining function.

***Insert Figure 7 Hereabouts ***

Since many words are activated at once, there is the potential for overactivation of constructions. For example since all words of category **cnoun** (count noun) have links to **cnoun-ness**, that node might receive more activation than appropriate, in cases when several count nouns are active (and of course this is the normal case, since FIG has both part-wise parallelism and competitive parallelism). Overactivation of **cnoun-ness** would in turn result in over-activation of, for example, **determinat**$_c$ and thus articles, which then could result in premature output of, say, *a*. Figure 7 illustrates the problem.

To deal with this FIG has "collector" nodes associated with syntactic categories and relations, such as **cnoun-ness** for **cnoun** and **purposer-a** for **purpose**$_r$. The activation level of collector nodes depends on the maximum (not the sum) of the activation received. For example, this "maximum rule" applies to the activation received by **cnoun-ness**, and so this node effectively ignores inputs from all but its most highly activated neighbor. Since **determinat**$_c$ receives activation via **cnoun-ness**, it also is effectively activated by one noun only. As another example, all fillers of the **possessor**$_r$ relation activate **genitive**$_c$, but, since the maximum rule applies to **possessor-a** FIG can produce, for example, *John's boy ate Mary's peach*, without the Genitive Construction becoming twice as highly activated as desired.

5 A Simple Example

***Insert Figure 8 Hereabouts ***

This section illustrates how various constructions participate in the production of a sentence, namely

the old woman went to a stream to wash clothes
glossing over only a few details (the full story is told in Ward (to appear)). For this example the input is the set of nodes **go**$_n$, **old-woman**$_n$, **wash-clothes**$_n$, **stream**$_n$, and **past**$_n$, linked together as shown in Figure 8. (In this section the names of the concepts are anglicized for the reader's convenience.) Boxes are drawn around nodes in the input so that they can be easily identified in subsequent diagrams. In fact this input is the result of parsing the Japanese sentence

obaasan wa kawa e sentaku ni ikimashita
old-woman TOPICN stream DEST wash-clothes PURPOSE go-POLITE-PAST
***Insert Figure 9 Hereabouts ***

Initially each node of the input is a source of activation. After activation flows, before any word is output, the most highly activated word node is **the**$_w$, primarily for the reasons shown in Figure 9. That is, thanks to activation from **subj-pred**$_c$, via the relations **volitional**$_r$, **responsible**$_r$, **part-cause**$_r$, **active**$_r$, **individuated**$_r$, and (negatively) **affected**$_r$, the most highly activated concept is the agent, namely **old-woman**$_n$. This concept activates the associated word, **old-woman**$_w$, which in turn activates **determinat**$_c$. The cursor of this construction being on its first constituent, this activation reaches **dt-1** and then **article**. From this, activation flows to **the**$_w$ and **a**$_w$. **the**$_w$ has more activation than **a**$_w$ because it also receives activation from **topic**$_n$.

*** Insert Figure 10 Hereabouts ***

Figure 10 summarizes the state of the network at this point. The nodes are listed by type and ordered by amount of activation. The position of the cursor of each construction is shown by capitalizing the relevant constituent. The highly activated nodes **obaa**$_n$ and **iku**$_n$ at the lower left are the ones that are being anglicized as **go**$_n$ and **old-woman**$_n$, respectively.

After *the* is emitted the update mechanism moves the cursor of **determinat**$_c$ to **dt-2**. The most highly activated word becomes **old-woman**$_w$ again, largely due to activation from **subj-pred**$_c$ via the "agent", **old-woman**$_n$.

After *old woman* is emitted **determinat**$_c$ is reset — that is, the cursor is set back to **dt-1**, and it thereby becomes ready to guide production of another noun phrase. Also, the cursor on **subj-pred**$_c$ advances to **sp-2**. As a result verbs, in particular **go**$_w$, become highly activated.

***Insert Figure 11 Hereabouts ***

go$_w$ is selected. Because **past**$_n$ has more activation than **present**$_n$, **infinitive**$_n$ and so on, **go**$_w$ is inflected (by a separate mechanism) and emitted as *went*. **motion-verb**$_c$'s cursor advances to its second constituent, which activates **destination**$_r$. (This construction exists exactly for this reason, to make the destination appear before the purpose for verbs like *come* and *go*.) **to1**$_w$ becomes the most highly activated word, due in part to to syntactic activation via **preposition** from the first constituent of **prep-phr**$_c$, which in turn receives its energy from **mvc-2**. Of all the prepositions in the network, **to1**$_w$ has the most activation since it receives semantic activation from **destination**$_r$, which receives its activation from **mvc-2**. This is shown in Figure 11.

After *to* is emitted, the cursor of **prep-phr**$_c$ is advanced. Moreover the relation **destination**$_r$ is marked as expressed, thanks to *to* being output. As a result, **stream**$_n$, the filler of **destination**$_r$, enters the focus of attention, which increases its activation level. The most important path of activation flow is now from **stream**$_n$ to **stream**$_w$ to **determinat**$_c$ to **article** to **a**$_w$. Thus **a**$_w$ is selected. The inflection mechanism produces *a*, not *an*, since **consnt-initial** is more highly activated than **vowel-initial**.

Then the cursor of **determinat**$_c$ advances and *stream* is emitted.

At this point the only node of the input not yet expressed is **wash-clothes**$_n$. This activates the relation it fills, namely **purpose**$_r$. Activation flows from this via **purposer-a** to **purpose**$_c$, thence to its first constituent **pc-1**, and thence to **to2**$_w$. After *to* is output the cursor advances, and *wash clothes* is output as a bare verb, thanks to the link from **pc-2** to **bare**$_n$.

Now that all the nodes of the input are expressed, FIG ends, having produced *the old woman went to a stream to wash clothes.*

6 Implications

There are some non-obvious implications of having many constructions active at once.

6.1 Synergy

***Insert Figure 12 Hereabouts ***

An important aspect of FIG is that the final output is determined by various constructions "working in synergy". A simple example of synergy is the cooperation of two constructions on the choice of a single word. For example, Figure 12 shows how a use of the word *make* is governed by **subj-pred**$_c$ and **per-causative**$_c$. It also shows that, for *eat*, the position is governed by **subj-pred**$_c$ and the form by **per-causative**$_c$. FIG is here in the spirit of Construction Grammar, where "an occurring linguistic expression [can] be seen as simultaneously instantiating more than one grammatical construction at the same level" (Fillmore 1988).

***Insert Figure 13 Hereabouts ***

Synergy also allows FIG to handle semi-frozen idioms. For example, FIG produces not only *Mary kicked the bucket* but also *Mary made John kick the bucket*, where the verb in the latter is bare. This is possible due to synergy between **kick-bucket**$_c$ and **per-causative**$_c$, as shown in Figure 13. The figure also shows that the order of *the* and *bucket* is determined in synergy with **determinat**$_c$.

Since all information is simultaneously available in FIG there is no need to explicitly move information around. This is in contrast with most generators, which have mechanisms to propagate information up and down trees, to pass information when making function calls, to make copies of information, to retrieve information when required for a syntactic decision, and so on. An example is the treatment of tense. Synergistic parallelism allows a construction like **per-causative**$_c$ to remain active over a period of time and affect the form of the verb in the embedded clause; there is no need for a mechanism to transport the information that 'the-verb-should-be-bare' down to the point of verb choice. Part-wise parallelism allows a node like **past**$_n$ to affect the inflection of the verb chosen whenever that choice happens to occur; that is, this information is available "globally" in the form of activation sent out from this node; there is no need for the syntax engine to invoke a procedure to retrieve the information governing tense from wherever it happens to be in the input.

***Insert Figure 14 Hereabouts ***

An advantage of using synergy is that constituents do not need to specify which constructions can instantiate them — that information is factored out. For example, the second constituent of **motion-verb**$_c$ simply specifies **destination**$_r$; thanks to synergy with **prep-phr**$_c$ the destination is expressed as a prepositional phrase, as shown in Figure 14. Similarly, the first constituent of **genitive**$_c$ specifies simply **noun**. By independent

principles (namely, the existence of **determinat**$_c$, **adj-mod**$_c$, and so on) the possessor is realized as a noun phrase which is as complex as is needed, for example, as in *a big old woman 's peach*. Similarly, in a purpose clause a full verb phrase can result even though only **verb** is specified explicitly by **purpose**$_c$.

***Insert Figure 15 Hereabouts ***

Since synergy allows information to be factored out into separate constructions in this way, FIG's constructions tend to be simple. FIgure 15 shows some fragments of FIG's knowledge network, showing how knowledge about determiners, intensifiers, adjectives, and count nouns is expressed in separate constructions. For example, the knowledge about the option of using an adjective is factored out into **adj-mod**$_c$. At run-time the information encoded in these constructions jointly determines the output.

Since each individual construction can be simple, each can have its own semantic motivation, as called for by functionalist approaches to grammar. That is, each construction can have its own specific source of semantic activation. For example, consider the construction **dir-particle**$_c$, which encodes the information that it is possible to use a verbal particle to express direction. This construction is linked to the node **motion**$_n$, and so whenever the input involves a motion, this construction will be activated and able to cause the interpolation of a verbal particle of direction after the verb.

The use of synergy also allows a grammar where constructions are suggested largely "bottom-up". For example, in FIG determiners are treated as being motivated by the presence of a count noun, rather than by the need to complete a maximal NP so as to satisfy the requirements of some higher-level construction. To be specific, there are paths from all count nouns via **cnoun-ness** to **determinat**$_c$. (Evidence that this is the right analysis is the fact that, even in the absence of any syntactic context, *a snake* or *the snake* seems more natural English than just *snake*.)

The idea of factoring out information into separate constructions and combining that information at run-time is not unique to FIG. For example, unification is a common way to combine information from different sources. The advantages of synergistic parallelism over other ways to combine syntactic information are that it is well suited to incremental generation and that it avoids the need to form binding lists or build up structures. These points are discussed further in §7.1.

For synergy to work, constructions must operate "in sync", that is, they must be organized with respect to each other. This is in part the responsibility of the syntactic update process (§3.4) and in part due to the weights on links. For example the verbal particle precedes the purpose clause, as seen in Figure 14, simply because of activation from **mvc-2** to **destination**$_r$; *once upon a time* appears at the beginning of sentences simply because the weights to and from **time-settng**$_c$ are relatively high; and the inner arguments of the verb precede adjuncts simply because the weights are set so that the former get more activation, in particular the weights to and from **transitive**$_c$ are relatively high.

6.2 Competition and Emergent Choice

This subsection illustrates how constructions affect competition among words and how constructions themselves compete.

***Insert Figure 15 Hereabouts ***

An example of competition occurs in the production of *John broke the dish* and *John made the dish vanish* from analogous inputs (they are analogous in that both have causative semantics). The reason that the causation is expressed implicitly in one case

and implicitly in the other is that certain verbs, including *break*, allow lexical causatives, whereas other verbs, including *vanish*, can only appear with a periphrastic causative. In FIG the possibilities are represented by **per-causative**$_c$, representing the Periphrastic Causative Construction, and **lex-causative**$_c$, for the Lexical Causative Construction. Certain verbs, including **break**$_w$, have a link to **lex-causative**$_c$. *** Insert Figure 16 hereabouts ***

After the causer of the event has been expressed, the cursors of both constructions advance. **per-causative**$_c$ activates the word **make**$_w$, and **lex-causative**$_c$ activates the category **verb**, as seen in Figure 16. The link weights are set such that **make**$_w$ is the default, but if **lex-causative**$_c$ is receiving activation (for example, from **break**$_w$) then the open-class verb (*break* in this example) will be selected. Thus the rivalry between these two constructions is played out as a competition between words. FIG never chooses one or the other construction explicitly — it simply doesn't need to. In this way the syntactic form of its utterances is emergent.

A similar type of competition occurs when the input only includes an action and its patient. In this case FIG will produce either an intransitive form, such as *the dish broke*, or a passive form, such as *the peach was eaten*, depending on the verb. The passive form is the default, but change-of-state verbs, such as *vanish* and *break*, have an alternative valence which allows an intransitive, patient-as-subject utterance (van Oosten 1985). In FIG there is accordingly a construction, **change-state**$_c$, which specifies that the verb can directly follow the patient. For certain inputs **passive**$_c$ and **change-state**$_c$ are in effect rivals; the rivalry is played out as competition between **is2**$_w$, activated by **passive**$_c$, and the open class verbs, which are activated by **change-state**$_c$. Weights are chosen so that the passive form is the default, but if **change-state**$_c$ is activated (due to activation received from a state-change verb) then a patient-as-subject utterance will be produced, for example, *John died*, instead of *John was died*.

Another case where competition among words results in alternative syntactic forms is the case of "optional constituents". By optional constituents I mean syntactic positions where the generator has the option of outputting a word, depending on whether the input includes suitable information. For example, pre-nominal adjectives appear if properties of an object need to be described. In FIG the availability of this option is represented by syntactic activation to **adjective** at the appropriate time. As a result adjectives get syntactic activation, and so to a lesser extent do nouns. There are two cases: If the input includes a concept linked (indirectly perhaps) to some adjective, that adjective will receive activation from that concept. In this case the adjective will receive more syntactic activation than any noun does, and hence have more total activation, so it will be selected next. Any number of adjectives can be emitted before a noun in this way. If the input does not include any concept linked to an adjective, then a noun will have more activation than any adjective (since only the noun receives semantic activation also), and so a noun will be selected next, and the adjective option will expire. In summary, in FIG, the decision to include or to omit an optional constituent (or adjunct) is emergent; for example, if an adjective becomes highly activated it will be chosen, in the usual fashion, otherwise some other word, most likely a noun, will be. This requires no additional mechanism; it occurs simply thanks to part-wise parallelism. This is in contrast to most generators, whose grammars include specifications of how to test the input to decide whether or not to use an optional constituent and what concept it will be used to express.

There is also competition as to which concepts are realized in which locations in the sentence, for example, in subject position. In FIG, **sp-1**, the first constituent of **subj-pred**$_c$, has links to various relations, as seen in Figure 5. The weights are chosen to

describe the "case profile" (Ward 1992a) of the ideal subject. Thanks to these links, the concept whose participatory properties best match the case profile currently being activated by constructions will receive the most activation. For subject this favors *John* for *John kissed Mary*; *Mary* for *Mary made the boy eat a peach*; topicalized concepts, such as *John* for *John was kissed by Mary*; *wind* for *the wind broke a dish*; and, for lack of anything better, *Mary*, for *Mary was killed* and *Mary died*. Thus the "slot-to-case" mapping is emergent in FIG; this is in contrast to most generators, which explicitly choose which concept to express in each syntactic role, typically relying on some kind of agency hierarchy, together with the rule that the highest ranking argument present takes the subject role, supplemented with a rule to override this based on focus and topic information, and so on. This approach also avoids the problems of defining a satisfactory set of deep cases.

In all these ways, and also in the determination of word order, as discussed in §3.4, FIG does without explicit choice among alternative syntactic organizations. Although an output may exhibit use of, for example, the Presentational Existential There Construction, this will be not because that construction was chosen over, say, the Subject-Predicate Construction; indeed, at run-time both of these constructions will have been active. Rather, the appearance of syntactic choice arises simply from the fact that the more highly activated construction has a greater effect on word choice. Thus constructions participate in language production but do not "exist" in the resulting utterances. In slogan form, linguistic knowledge exists in the speaker's head, not as part of the utterances he produces; and, in particular, constructions have constituents but utterances do not.

The de-emphasis of syntactic decisions sets FIG apart from most generators. It is common to include explicit syntactic choices among: which template to use, which pattern to execute, which arc to traverse, ways to syntactically realize a constituent, concepts to use in a grammatical role, possible orderings of words, and so on. Indeed, most generation research has followed McDonald (1983) in treating choice as the key problem in generation. However these abstract decisions are unnecessary (and in fact problematic (Ward to appear)).

The claim that FIG makes no explicit syntactic decisions has been challenged on the grounds that syntactic update is a kind of decision-making (Miezitis 1988). Syntactic update does involve decisions in the sense that it results in discrete changes in the state of constructions. Yet this update is more like monitoring something in the external world, namely the sounds or words that have been produced, than it is like making planning-type decisions. Moreover, the result of update in FIG is only a change in the levels of activation; whereas the hallmarks of more typical decisions are effects on the flow of control, for example, by triggering a branch or a subroutine call, or qualitative changes to the generator's state, for example by forming a variable binding.

The fact that syntactic choice is emergent is the reason FIG requires a separate update mechanism. Most generators simply choose a construction and "execute" it straightforwardly. However in FIG no construction is ever "in control". For example, one construction may be strongly activating a verb, but activation from other constructions may "interfere", causing an adverbial, for example, to be interpolated. Also, considerations having nothing to do with syntax, such as salience, may cause a word to be emitted, and then the generator must continue grammatically. For these reasons constructions need feedback on what words have been output.

7 Comparisons

This section discusses some common characteristics of treatments of syntax and discusses the extent to which they are present in FIG or should be present in general.

7.1 Structure-Building

Most generators build syntactic structures in the process of generation, unifying constructions or organizing them into syntax trees. (Not all generators, but those which do not build structures typically rely heavily on the structure of their inputs (§7.4).) FIG does not build up representations of syntactic structure; instead the network representations of linguistic knowledge affect word choice and word order directly.

This subsection explains the reasons for the pervasive use of syntactic structures and explains why these reasons do not apply to FIG.

It is often assumed that building up syntactic structures is simply a necessity. This is an instance of the more general assumption that for an organism to exhibit serial behavior it must have a representation of that behavior, some kind of plan, for example (Lashley 1951). It is certainly true that it is not easy to ensure that actions are performed in the right order, especially if the system must exhibit productivity, that is, if it needs to produce more than just perform fixed action sequences. However the standard answer, build representations, is not the only answer. This has also been pointed out in the context of perception, planning and acting by Agre and Chapman (1987) and Brooks (1991).

Another reason for structure building lies in the fact that generation research is largely parasitic on linguistic theories, despite the fact that mainstream linguistics is concerned with describing sentences in terms of structures, a rather different enterprise than building process models. (For historical perspective on the use of tree structures for syntactic description see Karlgren (1976) and Percival (1976).) It is worth noting that even in linguistics there has recently been some turn away from representations of structure. For example, Construction Grammar, among other modern theories of grammar, does without deep structure, that is, it has no intermediate level of structure between the meaning and the representation of surface structure. FIG goes one step further — it does not even represent surface structure.

Another reason for structure-building is that it goes hand-in-hand with reliance on explicit decisions. If a generator is to build up syntactic structures it must choose which constructions to incorporate, and if a generator is to make explicit choices it must save the results of these decisions somewhere. For example, if a generator chooses the concepts to express in various syntactic positions it must record that information, perhaps by binding variables; if a generator first chooses the words to say and then chooses their order, it requires some intermediate buffer in which to save the chosen but yet-unordered words; and if it assigns word order before actually emitting the words, it needs to assemble a representation of the sequence of words. FIG avoids the need for such storage by doing without explicit choice except for the current word.

To summarize the qualities which enable FIG to do without syntactic structures and explicit syntactic choice: synergistic and competitive parallelism mean that there is no need to bind constructions together; knowledge-source parallelism means that there can be synergy with semantic considerations; and the fact that it is incremental means that there is no need for a representation of word order. In the final analysis, since the purpose of speaking is to communicate, it should not be surprising that syntactic structure is at most a “side-effect”.

Doing without structures is good, not only for the sake of parsimony, but also because it eliminates overhead. This can be seen by comparing FIG to Finkler and Neuman's (1989) model which, although similarly inspired by the need for parallelism, retains the goal of assembling structures. As a result, their generator involves objects for syntactic structures getting acquainted, negotiating, and joining forces. By dispensing with syntactic structure-building, FIG avoids such complexity.

7.2 Capturing Generalizations

Generators using canned text or long, domain-specific patterns are inflexible. Thus there is a practical reason to factor out information into separate constructions (Jacobs 1985). This goal is also pursued under the slogan "capturing generalizations" in most approaches to syntax.

The goal of capturing generalizations has been a reason to posit syntactic structures underlying sentences. For example, it is possible to describe the similarity between an active sentence and a passive sentence with the same content words by referring to identity or similarly of representation at some "stage" of derivation. Generators based on such notions recapitulate generalizations at run-time every time they produce a sentence.

An alternative way to account for generalizations is in terms of the structure of the grammar — that is, the relationships among constructions (Lakoff 1987; Fillmore 1989; Jurafsky 1990). Inheritance is a common tool for capturing such generalizations (Jacobs 1985). An alternative to inheritance is to capture generalizations implicitly in the structure of connections in a PDP network; this appears especially useful for capturing sub-regularities. FIG, since it has very limited inheritance and is a structured connectionist system, simply fails to capture many generalizations.

One way that FIG does capture syntactic generalizations is by relegating them to semantics. This means it relies on activation flow via general (abstract) concept nodes, for example, any concept which has a link to $\mathbf{motion}_n$ will have access to the $\mathbf{dir\text{-}particle}_c$ construction.

The second way FIG captures generalizations is by factoring them out into separate constructions which are used together at run-time (§6.1). Such synergy may obviate the need for some inheritance, including multiple inheritance. For example, post-nominal modification, as in *the peach from the stream*, occurs thanks to synergy between $\mathbf{prep\text{-}phr}_c$, $\mathbf{determinat}_c$, and other constructions. In an inheritance based approach it might be necessary to instead have a separate 'Postmodified-np*' construction, existing only to inherit information from 'NP*', as was done in KING (Jacobs 1985). As another example, in FIG the "agent" of a passive is marked with *by* thanks to synergy with $\mathbf{prep\text{-}phr}_c$, whereas in KING this knowledge is applied indirectly, by means of a 'by-adjunct' pattern which inherits from 'prep-phrase'. Thus synergy allows two pieces of information to both affect the output directly; this contrasts with inheritance, which requires the information in one construction to be inherited and combined with another before it can be applied.

7.3 Grammaticality

Quite apart from theoretical motivations to focus on grammaticality, it is obvious that a generator's output ought to be grammatical rather than not. Yet many researchers ascribe supreme importance to this goal, believing that a generator should be guaranteed to produce grammatical output. This seems too strong a constraint; output which is grammatically correct is not necessarily more understandable than fragmented, ungrammatical

output. Thus, producing grammatical output was only one among several considerations when designing FIG. In fact, it would be very hard to prove that FIG's output is necessarily grammatical, as parallelism is rampant; even during the production of a simple sentence, many nodes have some degree of activation. FIG itself has no idea whether, when it has finished expressing some information, the result is grammatical or not. One could imagine improving the syntactic update mechanism to notice if some construction's cursor is in an odd state after generation completes, although working out the details would require some thought.

Also, to the extent that FIG's output is grammatical, it is partly because the input to be expressed constitutes a reasonable thought; syntax is not autonomous. Quite the contrary, syntactic knowledge only has the intended effects thanks to the information in the input, for example, for the sake of locality (§4.1).

It is not clear to me that incorporating a theory of grammaticality is important for process models in general, at least for a first pass. Doing without, as in FIG, enables things to be simpler in several ways. First, it allows FIG to do without "principles which govern the nesting and superimposition of constructions into or upon one another" (Fillmore 1988). Such principles allow a model to account for grammaticality judgements; a grammatical sentence is one where all the constructions "fit together snugly" (or unify without feature clash). In FIG constructions are active together, but are never compared, coordinated, matched, or bound together.

Second, doing without a theory of grammaticality allows FIG to do without an aspect of grammar required for fitting constructions together, namely a notion of external syntax, that is, for each construction, a description of "the larger syntactic contexts in which it is relevant" (Fillmore 1988). This requires otherwise unmotivated constituent labels for "non-terminals" such as 'N-bar'. These allow "upper" constructions like Subject-Predicate to refer explicitly to "lower" constructions like 'Noun-Phrase'; when these labels serve as tree nodes they are the points where the upper and lower constructions fit together. (If such constituent labels are available they can be used for other things also; for example, an 'n-bar' node can be used to ensure that determiners come before adjectives and that no noun phrase has two determiners. In FIG, word order information is simply encoded in link weights, and constraints of non-repetition arise due to the behavior of the syntactic update process.) In FIG a construction's privileges of occurrence are nowhere represented explicitly; rather this information is implicit in the links which point to it.

7.4 Constraints on the Input

Standard approaches to syntax for generation rely too narrowly on the structural characteristics of their inputs. Historically this is partly because meaning representations have been designed in the image of syntactic representations (Ward 1992c).

One aspect of this is the common assumption that inputs are hierarchical structures. An implication is that generator inputs usually have a privileged top node or entry point, the place from which traversal starts, corresponding to the S node of a syntax tree. The assumption of hierarchical structure is convenient because it allows generators an elegant control flow: traversing the input structure, and at non-terminals choosing a syntactic pattern with slots suitable for the child nodes, and proceeding recursively until the leaf nodes are found and mapped to words. Such "generation by recursive decomposition" is widely used (McDonald 1983; Hovy 1988) (although recently there has been a great deal of interest in more bottom-up, lexically-driven approaches (Kempen & Hoenkamp 1987; Finkler & Neumann 1989; De Smedt 1990; van Noord 1990).)

One reason is the perceived link between the productivity of language and compositionally structured representations of meaning, although in fact representations can be productive and even compositional without being traditional symbol structures, hierarchical or otherwise (van Gelder 1990).

Another example is the common assumption of conveniently labeled links among the elements of the input. Such links allow the generator to follow pointers in the input, for example to find the concept related by an 'agent' link to the current node. While this is clearly convenient for head-driven processing, labeled links are ontologically suspect (Ward 1992a).

Also common is the use of matching or unification of syntactic structures or word meanings to input structures. This presumes that inputs have particular structural configurations.

Relying on such details of the input representation makes a generator inflexible and, in particular, unable to handle inputs rich in information. FIG's generation "algorithm" is not controlled by the input, thus it is rather more lenient with respect to its input and consequently more robust, at least in principle. One application where this is clearly a good thing is machine translation (Ward 1989b). Existing systems are largely structure-bound, that is, the structure of the input closely governs that of the output. FIG is more flexible: there is no direct mapping from its input to its output. Thus it is more able to take advantage of the idiosyncrasies of the target language.

The only requirements that FIG places on its inputs are that they show: which concepts are involved, which pairs of concepts are related, and what are the participatory properties of each concept with respect to the larger meaning.

7.5 Ease of Representation

The formal or other properties of representations are often accorded primary importance in natural language processing. This paper, however, is about language processing; its claims involve the types of knowledge and computation required in generation. The specific representation used in FIG is merely a tool used allow exploration of these issues. Thus I have no interest in claiming, for example, that the knowledge representable in this approach includes all and only the true facts about possible human languages. Nor would I want to claim this representation can be used for other tasks, such as parsing. (Researchers who advocate the use of the same syntactic representations for both parsing and generation generally view both as simply tasks of mapping one representation onto another. To say anything about the use of shared knowledge for real language understanding and real generation will require a great deal more work, in my framework as in any other.)

I also do not want to propose FIG's representation as the one for linguists to use. In fact, FIG would be a terrible grammar-writing-and-testing tool for linguists. One reason is that in order to make FIG work it is necessary to choose link weights, a chore which is not always straightforward (Ward to appear). Another is that syntax in FIG is not autonomous, as discussed above. Perhaps the most important reason is that it is impossible to trace through execution by hand and predict what FIG will do, in part because of all the parallelism, and in part because syntactic choice is emergent and there is no structure-building. For example, in FIG the "adjective option" (§6.2) arises due to synergy among constructions; this complexity is resolved only at run-time. This contrasts with those approaches which use grammars where all the rules are "folded together", as in ATNs and systemic generators; in those systems options are explicit branch points, so one can inspect the grammar and readily determine all the sentence structures that can

result. This ability is probably not worth the price, however, namely the lack of synergy and the accompanying difficulty of generalization-capturing.

After all these disclaimers, there are some aspects of the representation of syntactic knowledge in FIG which do need to be discussed. One question is whether FIG's representation is easy to use. This seems to me to be the case — FIG's constructions are edited in a simple, readable format, not unlike those used in unification-based approaches — although ease of use is of course a subjective question. Another question is whether the representation is expressive enough. It seems to be, but it is possible that changes will become necessary as FIG is extended.

Another question is whether FIG is extensible. It would be nice to present a procedure for adding syntactic knowledge, including a set of principles for drawing links between nodes, such as, perhaps, "all constructions have links from all words which could be their heads". But of course, notions such as "head" are notoriously tricky; what counts as a head varies from grammatical theory to grammatical theory. Even if there were a perfectly satisfactory definition of "head" from a linguistic viewpoint, it might not be right for FIG; FIG's representation is subject to the further constraint that it work for processing. Perhaps after more experience encoding and debugging grammatical information it will be possible to present a discovery procedure or learning algorithm for extending FIG's network, but I have no confidence that linguistics can be done automatically, or even by humans following a fixed development methodology, at least in the near future.

7.6 Syntax in Other Connectionist Generators

This subsection compares the syntactic aspects of FIG to those of other parallel, mostly connectionist, approaches to generation.

Since connectionist models are inherently parallel, one might think that all connnectionist generators would exploit parallelism. When it comes to syntax, they don't. Previous structured connectionist treatments of syntax seem to be parallel only to the extent required by the connnectionist model of computation (Kalita & Shastri 1987; Gasser 1988). Indeed, they actually limit effective parallelism by using winner-take-all subnets and inhibition more generally. Thus these models are, despite the novel implementation, fairly conventional in design.

At the implementation level, however, there are interesting comparisons to make. Thus this subsection focuses on the extent to which the mechanisms and innovations found in FIG have been used or missed by builders of similar systems. The significance and shortcomings of other connectionist generation research are discussed more generally in Ward (to appear).

FIG's mechanism for updating the current syntactic state, the moving cursor, is not unique. Other connectionist generators have equivalent mechanisms: for example, Reich's (1970) nonterminal nodes sequence through a cycle of states, and Kalita has "sequencing nodes". Nor is FIG unique in resetting constructions, rather than making copies of them; both Kalita and Gasser have essentially the same mechanism. Nor is the locality principle unique to FIG — Gasser also uses flow of activation among the nodes of the input, although it is not whether it is used to deal with crosstalk. FIG's trick for handling optional constituents, using competition between words for an optional position and words that can follow it, is also used in Gasser.

The proposal that word order requires no mechanism other than that for word choice was first made by Stemberger (1985), but no previous implementation of this idea exists. Gasser uses the speed of activation flow to provide for word order, and Kitano (1990) uses

a separate concatenation operator to sequence words.

FIG is unique in using the maximum rule. As explained in §4.2 this is necessary to avoid overactivation of constructions, given that FIG has part-wise and competitive parallelism plus activation from words to constructions. That other connectionist generation researchers do not report problems of overactivation suggests that their systems have no significant part-wise or competitive parallelism.

FIG's treatment of the slot-to-case mapping as emergent is unique. To determine what concept to express in each "syntactic role" Gasser uses a clever mechanism involving simultaneous firing of nodes, and Kitano's system includes node names in the markers it passes around. Both effectively do structure-matching. To record the results of this process Gasser has a special role-binding mechanism; similarly Kalita has "binding nodes". When FIG used binding it led to problems (§8). Gasser and Kalita do not report such problems; I surmise that this is because their systems only had to produce a few outputs.

For Kitano syntactic knowledge is in large, very specific templates — that is, generalizations are not captured. Gasser has somewhat more general constructions, but they appear to be active one at a time, that is, there is no synergistic parallelism.

The idea of competing constructions has been proposed in the psycholinguistic literature (Baars 1980; Stemberger 1985), but choice among constructions is still resolved relatively early in these models, and in all implemented generators. For example, Gasser has explicit syntactic choice in the form of firings of nodes for constructions. Unique to FIG is the idea that construction "choice" is entirely emergent, that is, that competition among constructions continues even during word choice. This position is perhaps too radical; it is possible that a system with many more constructions will require some mechanism to ensure that after some point only a few compatible constructions have significant activation. Such a mechanism could perhaps simply consist of inhibitory links among constructions.

8 Some Personal History

The scheme proposed here is not the only conceivable way to handle syntax in an incremental, highly-parallel, word-centered, connectionist generator. However it seems that other approaches do not work as well; I know because I have tried several.

When building the earliest version of FIG I tried to do without explicit syntactic knowledge. I hoped that grammatical output would emerge from a very simple mechanism, namely transitional probabilities. That is, the likelihood of outputting a word of a certain syntactic category depended on the words just output; thus generation was a kind of Markov process. There was no other mechanism for representing syntactic knowledge. It was hard to make this approach work, and even harder to extend it, so, with reluctance, I considered more powerful ways to handle syntax.

My first refinement was to represent constructions explicitly in the network, with cursors to gate activation to constituents. In this version constructions were explicitly selected, just like words. Only after selection could a construction activate its constituents and affect word choice. Although only one construction could be active at a time, an activated construction could temporarily delegate this privilege to a subordinate construction. This meant that there was no longer any syntactic parallelism; this version was effectively a standard top-down generator where spreading activation only served to select constructions and to select words to use for the specified syntactic roles. Thus there was no synergistic and little part-wise parallelism, although this version did have competition

between optional constituents and those which could follow them.

I then realized that constructions need not be explicitly selected, that is, that syntactic choice could be emergent. In the next version therefore constructions were active in parallel. To organize and order constructions this version allowed nodes, especially those representing constructions, to send and receive names of other nodes, not just amounts of activation. Nodes were also allowed to have internal variables, these became bound to other nodes. For example, the internal variables of constructions were bound to concepts and words for constituents. This was thus a marker-passing or object-oriented system instead of a simple connectionist one. It was hard to make this scheme work. One problem was the need to make all possible bindings, or to decide which bindings to make — doing either was messy. Another problem was that, by binding nodes together, the system was creating links during the course of generation; it proved most difficult to tune link weights so that the network would settle as desired both before and after these links were added. This version of FIG handled the issue of multiple uses of constructions by actually making copies. For example, it would make a copy of $\mathbf{determinat}_c$ for each noun-expressible concept, and bind each copy to the appropriate concept, and to copies of $\mathbf{a}_w$ and $\mathbf{the}_w$. This also made the program hard to extend. In particular, it was hard to choose weights such that the network would behave properly both before and after new nodes were instantiated and linked in.

Finally I decided to dispense with structure-building and instantiation. Doing without binding concepts to constituents meant that crosstalk became a problem; to avoid this I gave FIG the locality principle. Doing without binding constructions together means that in this, the current version, word order is entirely due to amounts of activation. As I began to extend the system more rapidly it became more important to have a clean grammar, so I began to factor out more generalizations into separate constructions and to exploit synergistic parallelism. Exploiting synergy more also simplified the syntactic update process; for example, when there was a complex "noun-phase" construction it was necessary to label adjectives as "optional", and such optional constituents had to be treated differently for the purpose of syntactic update; this is not necessary in the current version. As I extended and tided the grammar the individual constructions began to look more like those familiar from Construction Grammar.

9 Coverage

FIG handles all the more obvious aspects of English syntax, and a slighter smaller fraction of Japanese, as suggested by the sample outputs shown in Tables 2 and 3. An enormous number of syntactic phenomena remain to be addressed; although of course this state of affairs is not unique to FIG.

*** Insert Tables 2 and 3 hereabouts ***

FIG currently has very few constructions. To extend it requires certain style of analysis of language phenomena, so it is good to know that there is a rich and growing vein of helpful work in the Construction Grammar tradition (Fillmore 1989)

Adding more constructions is of course not the whole story: one can't go very far without running into problems that cannot be handled by syntactic knowledge alone:

FIG needs knowledge about and mechanisms for prosody; this would enable it to handle syntactic phenomena such as end-weight, for example for heavy-NP movement.

FIG needs to be situated in a discourse situation and within a larger model of mind. In such a context it would be possible to have a distinction between given and new infor-

mation and between background and foreground information, which would enable FIG to produce anaphor and referring expressions more generally, and to handle some phenomena of ordering.

FIG requires detailed semantics for various domains, such as: scales, aspect, time, modality, questions, negative polarity, quantity, intrinsicness (for ordering adjectives), quantification, and space. As an example of the latter, FIG needs spatial reasoning ability and some notion of viewpoint for the sake of consistency, so it does not produce, for example, *the woman arrived at the stream and saw a peach going towards her*. It would certainly be possible to pad FIG's coverage by quickly throwing in more constructions, for example to make it produce questions, but this would be rather pointless until the semantics are understood. Thus there is a need for inference abilities for various semantic domains. It should be possible to do this with appropriate network structures, although I do not as yet have any proposals.

10 Conclusions

This paper has argued that the need for parallelism is pervasive in generation, presented a highly parallel approach to syntax for generation, and demonstrated its viability in the context of an incremental, word-centered, connectionist generator. This approach seems well suited to dealing with the complex inputs, and large knowledge bases likely for future generators — more constructions can simply be active in parallel, contributing to the scoring of more words.

However I have not yet demonstrated experimentally that this approach will in fact scale up. One reason for this is that there is currently no good source of information-rich inputs to test FIG on. Another is that there are currently no large inventories of lexical, syntactic, or world knowledge. Another reason is that FIG, as implemented, is not a very reliable system; it sometimes fails even on some simple inputs that it ought to handle. The main problem is that activation is being used for many things, and it is not easy to have the various uses not interfere with each other (Ward to appear).

Perhaps the most interesting result is the finding that syntactic structures and explicit syntactic choice are apparently are not necessary for generation. The computational properties of FIG which enable it to do without are: synergistic and competitive parallelism mean that there is no need to bind constructions together; knowledge-source parallelism means that there can be synergy with semantic considerations; and the fact that it is incremental means that there is no need for a representation of word order.

The significance of this negative result is not that practical generators which do build syntactic structures should be thrown away. Indeed, some of the ideas prototyped in FIG could probably be used to improve the performance of existing generators. However this result casts doubt on the value of continued research on intermediate representations for generation.

The needs of natural language processing, and generation in particular, have been the canonical illustration of the needs for symbolic processing and representation-building in cognitive science and artificial intelligence. But, since deliberate choice and the assembly of structures are not, in fact, needed for language generation (or for language understanding (Ward 1992b)), perhaps elsewhere also they are less important than has been thought.

If there is a methodological lesson to draw, it is that dogged experimentation with hand-built, eclectic, structured connectionist models can cast new light on old problems. Unfortunately, a current research trend is to re-implement traditional theories in con-

nectionist networks, either directly or by finding ways to implement representations of structures and variable bindings (Hinton 1990). This tendency is understandable; current connectionist learning procedures are limited, and it is not possible to build and train a PDP model of something complex without a fairly specific notion of what the network should do and how it should do it. People who want to build PDP models today are thus forced to borrow from elsewhere their notion of the nature of the task. For example, Miikkulainen (1991) accepted for his generator notions of deep case and the what-to-say/how-to-say-it distinction. The problem with this research strategy is, of course, that it sheds no new light on the nature of the tasks or the computation required to perform them. What seems to be needed instead is skepticism of old solutions and a willingness to experiment with alternatives.

Thanks to Daniel Jurafsky, Robert Wilensky, Jane Edwards, and Toshiaki Hisada. The work described here was largely performed at the University of California at Berkeley. It was supported in part by the Defense Advanced Research Projects Agency (DoD), monitored by the Space and Naval Warfare Systems Command under contracts N00039-84-C-0089 and N0039-88-C-0292; by the Office of Naval Research under contract N00014-89-J-3205; and by Mitsubishi Heavy Industries.

References

Agre, Philip E. & David Chapman (1987). Pengi: An Implementation of a Theory of Activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pp. 228–272.

Baars, Bernard K. (1980). The Competing Plans Hypothesis: an heuristic viewpoint on the causes of errors in speech. In Hans W. Dechert & Manfred Raupach, editors, *Temporal Variables in Speech*, pp. 39–49. Mouton.

Brooks, Rodney A. (1991). Intelligence Without Representation. *Artificial Intelligence*, 47:139–159.

Chafe, Wallace L. (1980). The Deployment of Consciousness in the Production of a Narrative. In Wallace L. Chafe, editor, *The Pear Stories*, pp. 9–49. Ablex.

Danlos, Laurence (1984). Conceptual and Linguistic Decisions in Generation. In *Proceedings 10th COLING*, pp. 501–504.

De Smedt, Koenrad J.M.J. (1990). Incremental Sentence Generation: a computer model of grammatical encoding. Technical Report 90-01, Nijmegen Institute for Cognition Research and Information Technology.

Feldman, Jerome A., Mark A. Fanty, Nigel H. Goddard, & Kenton J. Lynne (1988). Computing with Structured Connectionist Networks. *Communications of the ACM*, 31:170–187.

Fillmore, Charles J. (1988). The Mechanisms of "Construction Grammar". In *Berkeley Linguistics Society, Proceedings of the Fourteenth Annual Meeting*, pp. 35–55.

Fillmore, Charles J. (1989). On Grammatical Constructions. course notes, UC Berkeley Linguistics Department.

Finkler, Wolfgang & Günter Neumann (1989). POPEL-HOW: A Distributed Parallel Model for Incremental Natural Language Production with Feedback. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pp. 1518–1523.

Gasser, Michael (1988). A Connectionist Model of Sentence Generation in a First and Second Language. Technical Report UCLA-AI-88-13, University of California, Los Angeles.

Hinton, Geoffrey E. (1990). Preface to the Special Issue on Connectionist Symbol Processing. *Artificial Intelligence*, 46:1–4.

Hovy, Eduard (1988). *Generating Natural Language Under Pragmatic Constraints*. Lawrence Erlbaum Associates.

Jacobs, Paul S. (1985). *A Knowledge-Based Approach to Language Generation*. PhD thesis, University of California at Berkeley, Computer Science Division. also report UCB/CSD 86/254.

Jurafsky, Daniel (1990). Representing and Integrating Linguistic Knowledge. In *Proceedings 13th COLING*, pp. 199–204.

Kalita, Jugal & Lokendra Shastri (1987). Generation of Simple Sentences in English Using the Connectionist Model of Computation. In *Program of the Ninth Annual Conference of the Cognitive Science Society*, pp. 555–565. Lawrence Erlbaum Associates.

Karlgren, Hans (1976). Why Trees in Syntax? *Studia Linguistics*, XXX:1–33.

Kempen, Gerard & Edward Hoenkamp (1987). An Incremental Procedural Grammar for Sentence Formulation. *Cognitive Science*, 11:201–258.

Kitano, Hiroaki (1990). Parallel Incremental Sentence Production for a Model of Simultaneous Interpretation. In Robert Dale, Chris Mellish, & Michael Zock, editors, *Current Research in Natural Language Generation*, pp. 321–351. Academic Press.

Lakoff, George (1987). *Women, Fire, and Dangerous Things*. University of Chicago Press.

Lashley, Karl S. (1951). The Problem of Serial Order in Behavior. In L. A. Jeffress, editor, *Cerebral Mechanisms in Behavior*. John Wiley and Sons. reprinted in *The Neuropsychology of Lashley*, McGraw Hill, 1960.

McDonald, David D. (1983). Natural Language Generation as a Computational Problem. In Michael Brady & Robert Berwick, editors, *Computational Theories of Discourse*, pp. 209–255. MIT Press.

McDonald, David D. (1987). Natural-Language Generation. In Stuart C. Shapiro, editor, *Encylopedia of Artificial Intelligence, Volume 1*, pp. 642–655. Wiley.

Miezitis, Mara (1988). Generating Lexical Options by Matching in a Knowledge Base. Technical Report CSRI-217, Computer Systems Research Institute, University of Toronto.

Miikkulainen, Risto & Michael Dyer (1991). Natural Language Processing with Modular PDP Networks and Distributed Lexicon. *Cognitive Science*, 15:343–400.

Percival, W. Keith (1976). On the Historical Source of Immediate Constituent Analysis. In James D. McCawley, editor, *Syntax and Semantics 7: Notes from the Linguistic Underground*, pp. 229–242. Academic Press.

Reich, Peter A. (1970). *A Relational Network Model of Language Behavior*. PhD thesis, University of Michigan.

Stemberger, Joseph Paul (1985). An Interactive Activation Model of Language Production. In Andrew W. Ellis, editor, *Progress in the Psychology of Language, Volume 1*, pp. 143–186. Lawrence Erlbaum Associates.

van Gelder, Tim (1990). Compositionality: A Connectionist Variation on a Classical Theme. *Cognitive Science*, 14(3).

van Noord, Gertjan (1990). An Overview of Head-driven Bottom-up Generation. In Robert Dale, Chris Mellish, & Micheal Zock, editors, *Current Research in Natural Language Generation*, pp. 141–165. Academic Press.

van Oosten, Jeanne (1985). *The Nature of Subjects, Topics, and Agents: A Cognitive Explanation*. Indiana University Linguistics Club, Bloomington, Indiana. revised version of a Ph.D. Thesis University of California at Berkeley, Linguistics Department.

Ward, Nigel (1988). Issues in Word Choice. In *Proceedings 12th COLING*, pp. 726–731.

Ward, Nigel (1989a). Capturing Intuitions about Human Language Production. In *Program of the Eleventh Annual Conference of the Cognitive Science Society*. Lawrence Erlbaum Associates.

Ward, Nigel (1989b). Towards Natural Machine Translation. Technical Report AI89-30, Institute of Electronics, Information, and Communication Engineers Artificial Intelligence Research Group, Tokyo.

Ward, Nigel (1992a). An Alternative to Deep Case for Representing Relational Information. In *Proceedings 14th COLING*.

Ward, Nigel (1992b). An Evidential Model of Syntax for Understanding. Technical Report 88-3, Information Processing Society of Japan, Natural Language Working Group, Tokyo.

Ward, Nigel (1992c). Some Neglected Aspects of the Generation Task. *Computational Intelligence*, 8(1).

Ward, Nigel (to appear). *A Connectionist Language Generator*. Ablex. revised and extended version of A Flexible, Parallel Model of Natural Language Generation, Ph. D. thesis and Technical Report UCB–CSD 91/629, Computer Science Division, University of California at Berkeley (this also incorporates the material in all my papers except An Evidential Model . . .).

Figure 1: FIG Control Flow

Figure 2: A Tiny Fraction of FIG's Knowledge

Figure 3: The Presentational Existential There Construction as a fragment of the network

Figure 4: The Purpose Clause Construction

Figure 5: The Subject-Predicate Construction

Figure 6: The Determination and Genitive Constructions. The dotted line represents a link which does not pass activation; but which is used for syntactic update (§3.4)

Figure 7: The high fan-in to **cnoun-ness**

Figure 8: An input to FIG. %**agent**$_r$ is actually not a single actual relation, but an abbreviation for the list **volitional**$_r$, **responsible**$_r$, **partial-cause**$_r$, **active**$_r$, **individuated**$_r$, **topic**$_r$, **direct-cause**$_r$ and (negatively) **affected**$_r$ (Ward 1992a).

Figure 9: Selected paths of activation flow just before output of *the*

Figure 10: Activation levels of selected nodes just before output of *the*

Figure 11: Selected paths of activation flow just before output of *to*

Figure 12: Selected constructions affecting generation of *Mary made John eat a peach*, drawn to suggest the way constructions affect word choice during production. For clarity **subj-pred**$_c$ is drawn twice, although there is only one **subj-pred**$_c$ node in FIG; reuse of constructions is discussed in §4.2.

Figure 13: Selected constructions affecting generation of *Mary made John kick the bucket*

Figure 14: Selected constructions affecting generation of *John went to Chicago to kiss Mary*

Figure 15: Selected constructions affecting the generation of *John's very big peach*

Figure 16: Sources of activation to **break**$_w$, in the course of producing *John broke a dish*