Generating Text from Compressed Input: An Intelligent Interface for People with Severe Motor Impairments

PATRICK W. DEMASCO and KATHLEEN F. MCCOY

Appears in *Communications of the ACM*, May 1992, Vol. 35, No.5

Computers and computer-based technology have become an integral part of the lives of many individuals with disabilities. One of the most common activities that can be computer assisted is the generation of text. People who cannot accurately control their extremities (due to disabilities such as cerebral palsy and spinal cord injury) use computers as writing tools. People whose physical disability restricts their spoken output may use a computer as a communication prosthesis. In both cases, the generation of text is a necessary activity that can be physically demanding. It should be made as easy for the user as possible. While the standard computer keyboard is an efficient interface for able-bodied people and some disabled people, it may present significant access problems for others. In these cases some alternative interface is necessary.

Virtual Keyboards

Alternative interfaces for generating text, which we call *virtual keyboards*, can be modeled as having two components: a *physical interface* which accesses elements from a *language set*. The physical interface consists of the sensors and/or devices that the user physically interacts with (e.g., switch, joystick). The language set is a structured collection of linguistic units (e.g., letters, words, phrases) that the user selects from. The virtual keyboard is functionally defined by the mapping of physical input to selections from the language set. For example, the standard computer keyboard can be modeled as an array of switches, and a set of characters. Each character is selected by a single switch press (e.g., “a”) or a combination of switch presses (e.g., “Shift” + “5” = “%”).

The virtual keyboard model allows us to combine a wide variety of physical devices and language sets to form an interface ideally suited to the specific abilities of the user. For example, many individuals with motor impairments only have the ability to control a single switch input. The specific device can be anything from a large paddle switch to a sensor that monitors eye blink. For these individuals, a technique called *scanning* is commonly used. In scanning, the vocabulary set may be presented to the user on a dynamic display. The user makes selections by responding (i.e., hitting a switch) to a visual cursor that advances through the vocabulary set at an appropriate rate. Figure 1 shows the display associated with the row-column variety of scanning.

First, each row is sequentially highlighted. When the highlight arrives at the desired row, the user hits the switch (Fig. 1(a)). The highlight then advances across each item in that row until the user hits the switch again (Fig. 1(b)) selecting the currently highlighted item.

_	E	A	H	L	G
T	O	N	U	M	P
I	S	D	C	V	0
R	Y	B	J	1	2
W	F	X	3	4	5
K	Z	6	7	8	9
Q	.	<-	<w	Shft	Menu

(a) First Step: row scan

_	E	A	H	L	G
T	O	N	U	M	P
I	S	D	C	V	0
R	Y	B	J	1	2
W	F	X	3	4	5
K	Z	6	7	8	9
Q	.	<-	<w	Shft	Menu

(b) Second Step: column scan

Figure 1. Row-column scanning: A Virtual Keyboard for single switch input

Notice that in the lower right hand corner of the scanning array, there are a number of function keys. “<-” and “<w” are delete (character and word) keys. “Shift” functions similarly to a standard shift key, but it is selected in sequence (“shift” **then** “character”) rather than in a chorded fashion (“shift” **and** “character”). Finally, the “menu” key allows the user to branch to other pages within the vocabulary set (e.g., common words and phrases). This facilitates the development of large vocabulary sets that can be more efficient for the user.

There are a number of observations that can be made about Figure 1 and about the scanning technique in general:

1. While row-column scanning requires two switch activations for every selection, it is typically more efficient than linear scanning (one switch activation) because, on the average, access time to an individual unit is much shorter.
2. The access time to any individual unit varies according to its distance from the origin. Accessing the top-leftmost item is much more efficient than the lower-rightmost item.
3. The arrangement in Figure 1 exploits the differences in access time by placing the most frequently occurring characters in the most easily accessed locations. For spelling, this increases the efficiency of communication by a factor of two. One should note that such an increase in efficiency is accompanied by an increase in cognitive load. Individual items are initially more difficult to find when the set is sorted by frequency.

A Word-Based Virtual Keyboard

The virtual keyboard shown in Figure 1 is quite similar to a standard keyboard in that the user is constructing messages by spelling. Many assistive devices used to generate text take advantage of the power of the virtual keyboard model by allowing more direct access to vocabulary at the word level. This has two distinct advantages. First, a user will be able to access many words with fewer actions than with a letter-based keyboard. This can potentially improve their overall typing rate. Second, many severely physically disabled individuals also have language delays that make spelling difficult or impossible. By providing whole word access via orthographic or pictorial word representations, it is possible to make systems accessible to many more disabled individuals.

Figure 2. shows a simplified virtual keyboard based on whole word access.The user first selects a word category from a top-level display (Figure 2(a)). The display then presents all of the words associated with that category and the user selects the desired word. For example, to select the word family, the user first selects the *people* category and then selects the word *family*.

action	animal	body	clothing
color	emotion	food	heath
home	leisure	name	nature
$\boxed{\text{people}}$	place	relation	school
social	time	vehicle	work

(a) Top Level Categories

baby	boy	child	class
daughter	dentist	doctor	everybody
family	farmer	father	friend
girl	grandfather	grandmother	he
him	her	I	me
man	mother	nobody	nurse
people	person	police	priest
she	somebody	son	student
teacher	team	them	you

(b) People Category

Figure 2. Simplified Word-Based Virtual Keyboard. The selection of people from the Top Level Categories (a) results in the display of the words associated with that category in (b).

With this type of virtual keyboard, it is possible to offer the user a fairly large vocabulary that can be accessed with just two selections. For example, if we had 72 categories arranged in an 8 by 9 matrix, the user could access a vocabulary of over 5000 words with just two selections (one selection for the category and one selection for the word). The actual design of a word-based virtual keyboard is a highly individualized process where a clinician must establish a relevant vocabulary for the user and organize it in a way that is both efficient and relatively easy to learn and retain over time [Brandenburg & Vanderheiden 88].

In this paper we concentrate on improvements that can be made with a virtual keyboard using a word-based language component.

Current Research Directions

Current research efforts are based largely on the goals of developing more efficient virtual keyboards and extending the accessibility of systems to people with communication impairments. This has largely been accomplished by augmenting the virtual keyboard model with additional processing. The processing necessary to implement the virtual keyboards described previously is simply the mapping of a user's physical actions (e.g., selection) to a static data structure called the vocabulary set. The result of selecting an item may be the return of the item itself (as in the letter based example). A selection may also trigger the display of a new page of items (as in the selection of a word category). The two major enhancements to this model have been the addition of predictive capabilities to make the vocabulary set a dynamic structure, and the development of coding strategies to limit the number of selections necessary to access a particular item while maintaining a large vocabulary coverage.

Word prediction is one method for rapidly accessing words from a large dictionary [Gibler 83], [Arnott 84], [Hunnicutt 90]. Typically, word prediction systems consist of a virtual keyboard that has both letters and words. The words on the display are dynamically updated based on previous input. Predictions are based on word-initial letter input, word frequency, and, in some cases, previous words in the utterance [Swiffin 87]. While word prediction systems are useful for a large number of users, such use requires some spelling skills, and thus may not be appropriate for some individuals.

Abbreviation Expansion systems are another method of producing words with relatively few keystrokes [Vanderheiden 84]. Typically such systems are letter based, but allow the user to select words by inputting a pre-defined abbreviation of the word. While this method allows the user to compose words with few keystrokes, it requires the user to memorize a large set of codes for accessing the words and thus places quite a cognitive burden on the user.

Semantic Coding is another type of coding system which gives the user access a large body of words. In this method, the vocabulary set consists of semantic primitives that can be sequenced to form words. Minspeak™ is a system that represents semantic concepts as icons. Each icon can have multiple meanings depending on its context [Baker 82]. A set of approximately 100 icons can be combined in a variety of two- to three-icon sequences to generate a large number of words.While this technique is very useful, it does require the user to memorize a large number of sequences (on the other hand, because the codes are semantically based, they are presumably easier for the user to memorize). In addition, modified forms of root words (e.g., plural) are difficult to represent semantically and have to be modified by explicit "morphology" keys.

Sentence Compansion

Our work expands the virtual keyboard model in a different way than either prediction or coding. Assume that the user has access to a word-based language component through some appropriate physical device. The technique that we have developed, called sentence compansion [Demasco et al. 89], [McCoy et al. 90], takes a compressed message and expands that message into a well-formed sentence. The intent is to speed the communication rate of an individual intending to generate well-formed sentences. From a practical standpoint, this technique is most useful as a writing tool or as a conversational aid in settings where grammatical output is desired. The system is based on a virtual keyboard whose vocabulary set primarily consists of uninflected content words. The technique allows the individual to select the uninflected content words of their desired sentence, and yet have the device output a syntactically well-formed sentence. The system can speed communication rate in two ways:

1. Because the vocabulary set is limited to uninflected content words, the number of items that must be accessed in making a selection is reduced thus decreasing the amount of time necessary to select an item [Rosen & Trepagnier 82],
2. Because the user must only select the content words of the desired sentence, the number of selections required is reduced thus decreasing the amount of time required to construct the sentence.

The target population for the system described is comprised of individuals with severe physical limitations who are cognitively intact and linguistically proficient. Such individuals would benefit from the use of a system that produces grammatical output in a reasonable amount of time [Kraat

84]. In addition, we also envision that the system could be useful for individuals with some language impairment (the ability to recognize and sequence words in a meaningful way). In this case compansion may function more as a language assist. Finally, another possible application is as a clinical training aid for people with more severe language deficits. The system we describe will expand compressed input into the desired well-formed sentence. At the same time, the system should place a minimal burden on the user. Thus, we are not interested in a simple coding (cf. [Baker 82]) system where sentences have been stored and are simply indexed by their content words.

Input to our system are the uninflected content words of a sentence the user wishes to convey, for instance, “APPLE EAT JOHN”. In order for the system to generate a well-formed sentence, it employs a *semantic parser* to form a semantic representation of the input words. In this example, the parser must recognize that EAT can be a verb which accepts an animate ACTOR and an inanimate/food THEME in order to correctly infer the semantic relationships between these input words. The resulting semantic representation (along with a specification of the original word order) is then passed to the *translation component* which is responsible for replacing the semantic terms with their language-specific instantiations. The final phase of the processing is a *sentence generator* which attempts to form a syntactically correct sentence that retains the general order of the original input words producing, for example, “THE APPLE IS EATEN BY JOHN”. In cases where the user’s original input is ambiguous or the word order cannot be maintained, the system may present the user with several choices from which the user may choice his/her desired utterance with an additional selection.[1] Note that if the above example was input from a word based virtual keyboard system which requires an equal number of keystrokes per input word, our

1. The exact nature of this aspect of the interface design is beyond the scope of this work.

system would yield a keystroke savings of 50%. A diagram for the system processing is shown in Figure 3.

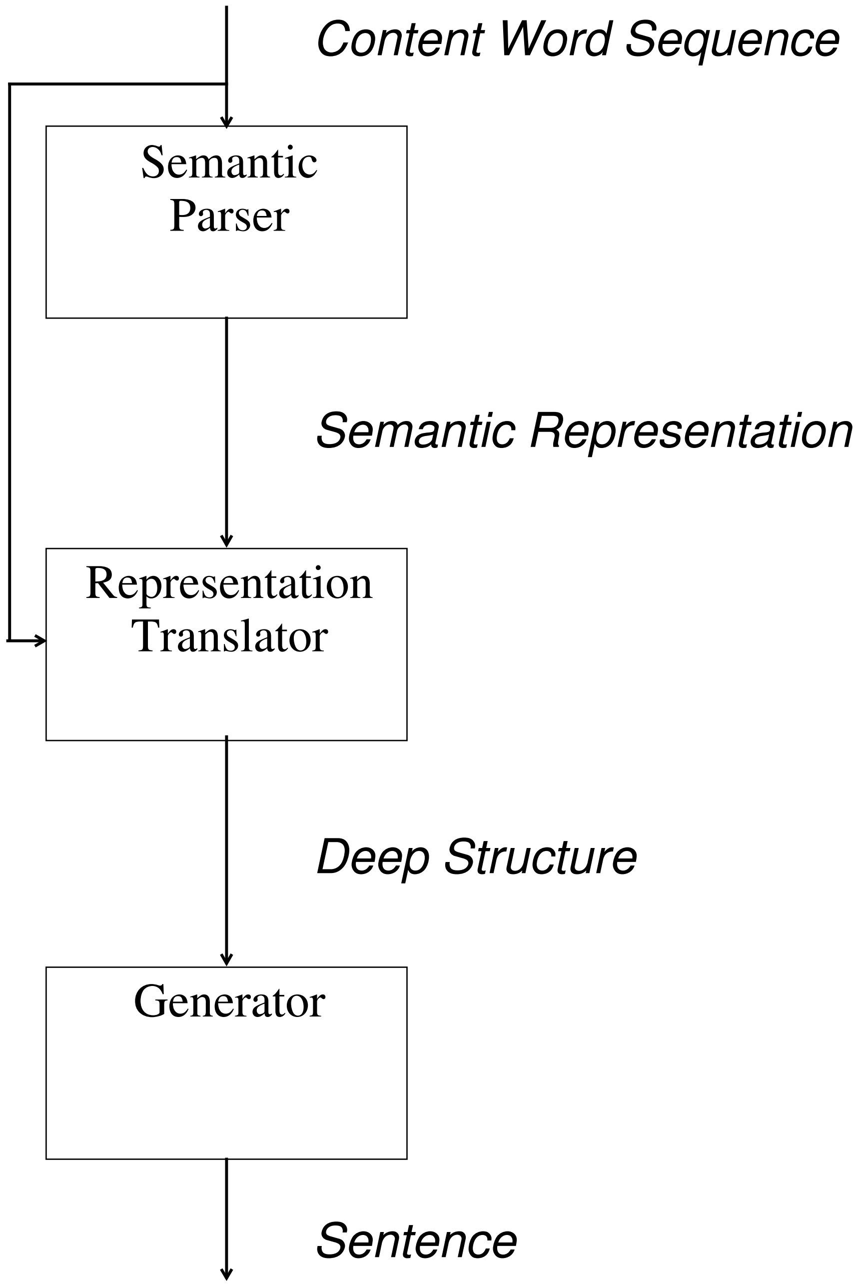

Figure 3. Module Structure for the Compansion System

Semantic Parser

In order to generate a syntactically well-formed sentence, the input must be understood to some extent so that only meaningful sentences are generated. For example, for the input “APPLE EAT JOHN” we do not want the system to produce “THE APPLE ATE JOHN” because that is not a semantically appropriate sentence.

Thus, the sentence compansion system employs a semantic parser to generate a meaningful semantic representation which captures how the input content words fit together. The semantic representation that we use is a case frame representation based on a variation of *case grammar* which was originally introduced in [Fillmore 68], [Fillmore 77]. In a case frame representation of a sentence, the verb is identified and the noun phrases in the sentence are said to play a role with respect to the verb. The number of roles which a noun phrase can play are small. For example, three common roles are ACTOR (the object doing the action), THEME (the object being acted upon), and LOCATION (the place where the action is done). Such a case frame representation is typically the kind of representation which is the output of a “semantic processor” of a natural language understanding system (see [Allen 87], [Winograd 83] for an introduction to natural language processing techniques, and [Palmer 84], [Allen 87], [Hirst 87] for more information on semantic representation and processing).

Traditional natural language processing systems break the process of generating a semantic representation from a set of input words into two phases ([Allen 87], [Winograd 83]). First, a syntactic phase uses a grammar of English to generate a syntactic parse of the input sentence. The syntactic phase assumes that the sentence is well-formed with respect to the grammar. Its output is a syntactic parse tree that indicates the syntactic categories of each word in the sentence (e.g., noun, verb, adjective) along with information concerning which parts of the input modify which others. The second phase of processing, the semantic phase, takes the parse tree and matches it against a set of semantic interpretation rules (typically associated with individual lexical items) to generate the semantic interpretation. For instance [Allen 87, Page 233] shows a set of rules associated with the verb break. A typical rule says that if the SUBJ of the sentence is an animate object and the OBJECT is inanimate (but physical) then the SUBJ is the ACTOR and the OBJECT is the THEME. This rule would fire in a sentence such as “JOHN BROKE THE WINDOW” resulting in JOHN being marked as the ACTOR and WINDOW being marked as the THEME.

While the output of our semantic parser is the same as for the semantic phase of a traditional natural language understanding system, our work may not rely on this traditional method because our expected input is severely ill-formed with respect to English grammar. It contains only content words so all determiners, prepositions, and word inflections are not given. In fact, our input is so ill-formed that we cannot benefit from staying within this paradigm even if it is augmented with mechanisms that handle some syntactic ill-formedness (cf., [Granger 83], [Fass & Wilks 83], [Weischedel & Sondheimer 83], [Jensen et al. 83], [Carbonell & Haynes 83], [Milne 86]).

There have been some natural language processing systems that rely on very little syntactic regularity of the input. In order to generate a reasonable interpretation, however, the system can only work within a very restricted semantic domain (e.g.,sub-language [Marsh & Sager 82], [Marsh 83], [Marsh 84]). Our system may not draw on this work, however, because we are faced with an unrestricted semantic domain.

Our semantic parser is faced with a syntactically-degraded input from a semantically unrestricted domain. Following [Small & Reiger 82], we rely (in a bottom up fashion) on semantic information associated with individual input words and allow the words to mutually constrain each other in order to form a well-formed sentence. We have added a top-down processing component in order to further reduce ambiguity.

The parser must attempt to generate a semantic representation for the input words. Recall that in our semantic representation it is important to determine the syntactic category of each word of input (e.g., noun, verb, adjective) because the representation revolves around the verb with each noun phrase playing one of a fixed set of roles with respect to the verb. Both the verb and any of the noun phrases may be modified (by an adjective or adverb, for example).

Each word in the system is thus associated with the syntactic category that the word can belong to in a valid English sentence. The categories we use are: verbs, objects (for any noun), adjectives, and adverbs. Any given word may belong to more than one of these sets. For example, STUDY can be both a verb (as in "JOHN WILL STUDY") and an object (as in "JOHN IS READING IN THE STUDY").

Because these sets are used for a number of purposes, each is represented as a semantic taxonomy. For instance, the verb set is broken into Verbal, Relational, Material, and Mental following [Halliday 85]. The objects are divided into Physical and Abstract, with further divisions also represented (e.g., Animate vs Inanimate). Maintaining the various sets of words in a hierarchical fashion allows us to save space since information cab be placed in the hierarchy and then be inherited by a number of different words. Thus, the information does not need to be repeated for each individual word.

The initial phase of processing within the semantic parser creates (at least one) subframe for each input word. The subframe contains the word itself along with its type information from the various taxonomies. For example a frame will be created indicating that John is an Animate-Object from the object taxonomy. Any given word will be given type information from each instance in the taxonomies (e.g., STUDY will occur at least twice, once from the object taxonomy as a physical-location and once as a verb).

Once the subframes have been created, the system must determine how the individual subframes can fit together in a meaningful way to form the semantic interpretation of the desired sentence. For each word (subframe), there are three possibilities:

1. it is the main verb of the sentence,[2]
2. it is a noun (phrase) which is playing one of the fixed set of roles with respect to the verb,
3. it is a word modifying a word in one of the above two possibilities. I.e., it is an adjective modifying a noun or an adverb.

The parser first attempts to attach all words in (3) above (i.e., all adverbs and adjectives) to the word they are modifying. Under the principle that a certain adjective or adverb can only modify certain types of other words, the adjective and adverb taxonomies contain links to the types of all other words that they can possibly modify. For example, the adjective "BIG" is linked to the type PHYSICAL-OBJ and not to type ABSTRACT-CONCEPT in the OBJECT taxonomy. This link tells us that in the input "JOHN STUDY WEATHER BIG UNIVERSITY", the word "BIG" can either modify the word "UNIVERSITY" or the word "JOHN" since both are PHYSICAL -OBJs. "BIG" cannot modify the word "WEATHER" since it is not.[3]

These links are used in a bottom-up fashion to combine the individual word sub-frames into larger sub-frames. Since, for example, the PHYSICAL-OBJ "UNIVERSITY" can be modified by the adjective "BIG", the initial subframes for these two words will be combined into a larger subframe which takes on the semantic properties of the word being modified (in this case, UNIVERSITY). Thus the adjective will not be seen by further processing. The frame type of the resulting frame will be Physical-Place (because of the word "UNIVERSITY"). Inside the frame (hidden by the semantic parser) will be the information stating that "UNIVERSITY" is being modified by the adjective "BIG".

2. The system assumes only one verb per sentence.
3. Our system is not designed to handle non-literal uses of language.

At this point in the processing the semantic parser has a number of possible subframes for each word of the input. We are now at a juncture that could benefit from some top-down direction in order to determine which subframe to choose for each word and to determine what role each noun is playing with respect to the verb. Notice that the main verb of a sentence predicts much of the structure of the overall sentence. Knowing the verb dictates which semantic roles are mandatory and which roles should never appear, as well as type information concerning possible fillers of each role. For example, the verb "go" cannot have a THEME case in the semantic structure. Furthermore, it cannot have a FROM-LOC case without having a TO-LOC at the same time. But "go" can take a TO-LOC without a FROM-LOC. The TO and FROM-LOCs must be physical locations. Each verb of the system has a set of (uninstantiated) skeleton frames which contain typed variables where input words can be fit.[4] Each variable in the skeleton frame has associated with it a type that restricts the possible words that can instantiate the variable (i.e., the variable types are taken from one of the system knowledge taxonomies). Thus a skeleton frame indicates both which semantic roles can be filled in a legal sentence using that verb, and also indicates the types of words which may fill the indicated roles. For instance, one skeleton frame for the verb STUDY indicates that it must have an animate AGENT and a THEME which could be any kind of object. Another frame for the verb contains the same AGENT and THEME information, but additionally contains a LOCATION slot which must be filled with a physical location. Still other frames associated with STUDY allow other optional roles.

Since the top down information is associated with the main verb of a sentence, the main verb of the sentence must be identified in order to benefit from this information. If there is exactly one input word that can only play the verb role, then it is unambiguously taken to be the verb. If this is not the case, but several words of the input could be the main verb, processing is split and each possibility is tried. In practice, many of the possibilities are dropped because they cannot account for the rest of the input words. Finally, if none of the input words can play the role of verb, the verb must be inferred. In this case, the system infers the verb to be either BE or HAVE depending on characteristics of the input words. The choice of BE or HAVE as candidate verbs is motivated by their relational nature [Halliday 85], by their frequency of occurrence, and by informal experiments simulating the system we describe.

The result of choosing a main verb of the sentence will be the availability of a number of potential skeletons of semantic structure into which the sub-frames generated for the individual input words can be fit. The result of this top-down processing is a set of (partially filled) semantic structures. All well-formed structures (i.e., all structures whose roles have been filled and which have been able to accommodate each word of the input) are passed one at a time to the next components of the system.

Notice in our example (JOHN STUDY WEATHER BIG UNIVERSITY) that the input contains two words that could potentially play the verb role: STUDY and WEATHER (as in "the ship weathered the storm"). While WEATHER could be a verb, there are no input words of the correct semantic type to fill in its roles, so that interpretation is discarded. Thus the system is faced with the verb STUDY and its associated skeletons (one of which was described above) along with the sub-frames representing JOHN, WEATHER, and UNIVERSITY[5]. Because JOHN is the only word capable of playing the AGENT role, it is taken as the AGENT. Notice that both UNIVERSITY and WEATHER are capable of being the THEME of STUDY. However, UNIVERSITY is also capable of playing the LOCATION role while WEATHER is appropriate for no other role. Because each word of the input must be accounted for, the parser settles on the following interpretation:[6]

---

4. This information is actually spread throughout the taxonomy with verbs at lower levels inheriting the skeleton information from the higher levels.

5. Recall that the adjective is hidden from this part of processing since it has been incorporated into the sub-frame of the word(s) it can modify.

Semantic Parse: (ASSERT (VERB (LEX study))
(TIME PRES)
(AGENT (LEX JOHN))
(THEME (LEX weather))
(LOC (LEX university) (MOD big)))

Notice that in addition to information directly resulting from the input words, the semantic representation contains information which will be necessary for sentence generation (e.g., (TIME PRES)). Such information may be derivable from characteristics of the input words. In the absence of such information, the system relies on tense established in previous utterances.

Dictionary Translator

Before the actual generator can be entered, the case frame representation output from the semantic parser must be translated into the appropriate form. The use of a translator ensures modularity of the overall system by keeping the semantic parser and the generator independent of each other. Thus one can imagine using a slightly different semantic representation appropriate for a different application, and yet using the same generator component. The translator determines how each piece of the semantic structure can be realized syntactically. It is also responsible for extracting certain ordering information from the input string: the translator attempts to create an input to the generator that will eventually produce a sentence that maintains the original word order given by the user. Some of this ordering information is passed to the generator by specifying the "focus" of the sentence (i.e., that item which should appear first). Some modifier placement and attachment selections made by the translator may also affect final word order.

The design of the translator is based on work done by [McDonald 80] and [McKeown 85]. Each component in the case frame representation will have an entry in the "dictionary" which will hold its possible translations into linguistic structure. The translation process is complicated by the fact that the conversion of an individual element might be dependent on (the existence of) other semantic components. Thus the translation of an individual element relies, in general, on the entire semantic structure.

The Generator

The completed translation is passed to the generator component where a syntactically well-formed sentence is produced. While there are several sentence generation systems available today (e.g., [McDonald 80], [Meteer et al. 87],[Mann & Matthiessen 85]), we have chosen the functional unification model [Kay 79], [Kay 86], for the following reasons:

First, because of the nature of the functional unification grammar, the input to the generator can be in functional rather than syntactic terms (see [McKeown 85], [Appelt 83]). Thus the translator need not have sophisticated language dependent knowledge.

Second, a functional unification grammar makes it very easy to encode certain default information into the grammar. It is particularly useful to encode default lexical items for certain closed-class items which are unlikely to be specified in the input.

Finally, the availability, understandability, and demonstrated success of the functional unification grammar make it attractive. The actual unifier that our system employs was provided to us by Michael Elhadad and Kathy McKeown at Columbia University [Elhadad 88].[7]

---

6. The semantic parser also generates an interpretation where BIG modifies the word JOHN. This interpretation is not preferred by the system because it does not result in a sentence which maintains the user's original word order.
7. The grammar used by our system is our own.

The Functional Unification Model

A functional unification grammar describes the set of sentences in a language as sets of attribute value pairs. The particular attributes contained in the grammar are (for the most part) left open to the grammar writer, but may be syntactic, semantic, or functional in nature. The value of an attribute specifies legal fillers/values for that attribute. While the attribute-names are atomic symbols, the values may be either atomic symbols or sets of attribute-value pairs.

A grammar in this formalism is called an FD (functional description) and is a collection of attribute value pairs. The pattern attribute is special in that it defines the surface order of the FDs constituents in the final output string. FDs may also contain ALTs (alternatives) which indicate that a given category can have more than one construction. Thus, for example, the sentence grammar may be defined such that in a legal sentence, the verb may or may not take an object. The appropriate alternative may be chosen depending, for example, on features of the particular verb (encoded as an attribute value pair). In the functional unification model, both the input and the grammar are in the FD formalism. Viewing the grammar FD and the input FD together, it is helpful to think of the grammar FD containing "holes" where the lexical items from the input could fit and the input FD containing "holes" where the syntactic constraints from the grammar could fit. These "holes" are then filled by the process of unification.

Processing in the Model

The following sample input expresses the deep structure which would be input to the generator to produce the sentence "The apple is eaten by John." Note that it contains both functional categories (CAT, PROT, etc.) as well as the actual words (JOHN, EAT, APPLE).

Translation: ((CAT S)   (FOCUS GOAL)
(PROT ((NNP ((N === JOHN))) (PROPER YES)))
(VERB ((VVP ((V === EAT)))
(GOAL ((NNP ((N === APPLE))))))))

In this example CAT = S indicates that the category of the outer FD is a Sentence. The FOCUS attribute is a functional category which indicates that the focus of the sentence is the GOAL which is further specified later in the FD. This particular attribute appears in the grammar and is consistent with alternatives in the grammar that move the goal into a prominent position in the sentence (e.g., subject position). PROT =((NNP ((N === JOHN))) (PROPER YES)) indicates that the protagonist (actor) of the sentence is the noun portion of an NNP phrase and is represented by the lexicographic entry JOHN. It is further specified that the protagonist is a proper noun. The specification also indicates that the root of the verb is EAT, and the goal (theme) is APPLE.

Notice that several elements necessary to produce a sentence are missing from the input. For example, there is no information on the final order of the output words, correct endings to words, the person or number of the verb, whether determiners are needed, and, if so, where they go in the final output.

The input FD and the grammar FD are assembled together by unification into a single resultant FD. This final FD is then the formal representation of the semantics of the former and the syntax of the latter. During unification variables in the grammar are replaced by values from the input FD, and alternatives in the grammar are eliminated. The resulting intermediate form contains all of the information needed to generate an output sentence. Thus given the above input, the number feature on the verb will be correctly inferred from the number feature on the sentential subject of the sentence, in this case "apple", stored in the lexicon.[8] In addition, the missing article on "apple" will be inferred to be "the", since "the" is given as the default determiner in the grammar.

8. In our system the number feature is actually defaulted to singular. The lexicon only contains information when the default needs to be overridden.

In addition, the intermediate form will contain ordering information through the attribute "pattern" contained in the grammar.

Notice in this instance that the order of the final output is quite different from the order in the input FD. The passive sentence form is selected on the basis of the value of the focus attribute provided in the input. As a side effect of the choice of passive voice, the helping verb "is" and the preposition "by" have been added. In addition the main verb of the sentence is marked with the past participle form.

The next step in the model is to call the morphology function which adjusts the lexical entries in the intermediate form FD for the proper tense, number, etc. After morphology, the linearizer function is called to actually output the sentence. The linearizer takes the FD from morphology and follows the pattern of each FD to be sure that the lexical entries are all output in the correct order. During linearization the system has the ability to reject sentences that do not conform to the original word order input by the user. If no choices remain then this restriction is relaxed and all choices are given.

Evaluation

While the work described here is theoretical in nature and its usefulness to a particular user dependent on many factors (from the user's cognitive ability to the interface design) we have developed an evaluation of the potential effectiveness of the technique based on an analysis of text samples. By calculating the relative number of function words (e.g., articles) and words that are formed from roots and suffixes (e.g., plurals), we can develop estimates of selection savings over a comparable word-based system. We analyzed a 17,000 word text sample from the Carterette corpus [Carterette 74]. We chose to use the adult portion of the corpus (the entire corpus also contains transcripts of 1st, 3rd, and 5th grade conversations). Words in the sample were tagged according to their type. The types used were those that were relevant to the sentence compansion technique (function words and "root + suffix" words). Table 1 shows a summary of this analysis, occurrences of each word, and the right side contains the total number of occurrences for each type.

Word Type	Unique Occurrences		Total Occurrences	
	Number	%	Number	%
root	1347	71.0%	12,741	72.0%
root + "s"	260	13.0%	973	5.5%
root +"ed"	130	6.9%	705	4.0%
root + "ing"	112	5.9%	342	1.9%
root + "en"	9	.48%	53	.3%
prepositions	29	1.5%	1468	8.3%
conjunctions	3	.16%	646	3.6%
articles	3	.16%	828	4.7%
total	1893		17,756	

Table 1. Conversational Text: Word Frequency by type

Root Word Selection - In a letter-based system, the addition of a suffix represents a relatively small increase in typing burden. For example, “computers” requires only one additional keystroke over the word “computer”. However, in a word-based system, the inclusions of root words and their derived forms significantly increases the size of the vocabulary set and consequently increases the average access time to any individual unit. If the system did not require the user to input the derived forms of words, then the “+s”, “+ed”, “+ing” and “+en” words in Table 1 would not be necessary, and the size of the vocabulary set would be decreased by approximately 30% (i.e., the sum of the percentages of those four groups from the left side of the table). In a row-column scanning system, the average access time is proportional to the square root of the vocabulary set size [Rosen &Trepagnier 82]. A reduction of 30% in the size of the vocabulary set would result in a 15% reduction in access time.

Function Word Deletion - The second goal of the sentence compansion technique is to relieve the user from the burden of inputting function words. In Table 1, the function word categories of prepositions, conjunctions and articles comprise approximately 17% of all words in the sample. In a word-based system, this corresponds to a potential 17% keystroke savings if the sentence compansion technique is employed.

Defaults - Finally, although not shown in Table 1, we can also expect some rate enhancement through the use of defaults. If we assumed a default subject of “I” (e.g., “want car” -> “I want the car”) and a default verb of “have” or “be” (e.g., “John tired” -> “John was tired”), we could expect an additional 10% keystroke savings[9]. It is possible to have two default verbs because the semantic parser can infer the relationship between the subject and the object and correctly predict the likely verb.

With a 15% performance improvement from root word usage, a 17% improvement from function word deletion, and a 10% improvement from use of defaults, we can expect a maximum rate enhancement of approximately 42%. This estimate must be considered in relation to several factors. First, the initial estimates are perhaps low because they are given in relation to a “perfect” scanning system. That is, one that has been tailored to exactly what the user wants to say. Second, the performance improvement obtained from function word deletion must be tempered by the fact that commonly occurring function words such as articles would be made more accessible than other words. The actual time savings would then be less than the keystroke savings. Third, the 42% estimate assumes perfect accuracy in the technique. There will be many cases where the system will be unable to generate exactly one sentence from the user’s input. In this case, where the system is faced with ambiguities, the user will have to make another selection to choose his/her desired sentence. It is also feasible that the system will not be able to generate a sentence from the user’s input We hope that the rule-based nature of the system will result in errors that are predictable to the user. In cases where the user anticipated an error in interpretation, he/she could simply enter a less compressed sentence.

Current Status and Future Directions

The current compansion model is implemented on a Sun SPARC Workstation using Common Lisp. It has a dictionary of approximately 2000 words and attempts to handle unknown words by inferring the likely case role that should be filled for a given semantic representation. It also allows the user to omit the sentence subject (filled in by the default “I”) and/or the sentence verb (filled in by the defaults “be” or “have” depending on the sentence object) for additional rate enhancement. Our current work is primarily focused on expanding the range of sentence structures that the system can process (e.g., embedded clauses).

9. The system permits both “have” and “be” as defaults and decides which verb is most appropriate, based on other case roles in the sentence. For example, “I happy” would be expanded into “I am happy” while “I book” would be expanded into “I have a book”.

While the theoretical evaluation of the sentence compansion technique shows great promise, its general utility must be evaluated with respect to individual user populations with specific physical interfaces. One such effort is underway. We are collaborating with the Prentke Romich and Semantic Compaction Companies to develop a version of the sentence compansion system with a suitable user interface to run on currently available augmentative communication hardware. The effort includes using a iconic interface based on Minspeak for word selection with a simplified version of the sentence compansion system in order to produce well-formed sentences. The completion of the project will result in a system which can be tested with various user populations and tailored to their individual needs. We view our current efforts with a Lisp-based prototype as a necessary predecessor to the development of such practical systems.

Conclusion

We feel that the sentence compansion technique has great promise for word-based virtual keyboards. In addition to making text entry more efficient for many individuals, this technique has the potential to assist people with language deficits in sentence formation. Furthermore, some of the underlying technology behind this system may be extended to other interface techniques. For example, the semantic parser might be a useful component to an improved word prediction system. Through these efforts, we hope to make progress towards the development of more intelligent interfaces that shifts much of the physical burden from the disabled user to the assistive device.

Acknowledgments

This work is supported by Grant Number H133E80015 from the National Institute on Disability and Rehabilitation Research. Additional support has been provided by the Nemours Foundation. We would like to thank Yu Gong, Mark Jones, Chris Pennington, and Charles Rowe for their efforts in both the design and implementation of this system.

References

[Allen 87] James Allen. *Natural Language Understanding*. Benjamin/Cummings, CA, 1987

[Appelt 83] D. Appelt. Telegram: a grammar formalism for language planning. In *Proc. 21st Annual Meeting of the ACL*, pages 74-78, Assoc. Comp. Ling., Cambridge MA, June 1983.

[Arnott 84] J.L. Arnott, J.A. Pickering, A.L. *An adaptive and predictive communication aid for the disabled that exploits the redundancy in natural language.* In RESNA 7th annual conference, pages 349-350, RESNA, Ottawa, Canada, 1984

[Baker 82] B. Baker. Minspeak. *Byte*, 186ff, September 1982.

[Brandenburg & Vanderheiden 88] Sara Brandenburg and Gregg C. Vanderheiden. Communication Board Design and Vocabulary Selection. In *The Vocally Impaired: Clinical Practice and Research*. pages 84-133, NY: Grune and Stratton, 1988

[Carbonell & Hayes 83] Jaime G. Carbonell and Philip J. Hayes. Recovery strategies for parsing extragrammatical language. *American Journal of Computational Linguistics*}, 9(3-4):123-146, 1983.

[Demasco et al. 89] Patrick Demasco, Kathleen F. McCoy, Yu Gong, Christopher Pennington, Charles Rowe. *Towards More Intelligent AAC Interfaces: The Use of Natural Language Processing*, In RESNA 12th Annual Conference, New Orleans, LA, 1989.

[Elhadad 88] Elhadad, M.*The FUF Functional Unifier: User's manual*. Technical Report # CUCS-408-88 Columbia University, 1988.

[Fass & Wilks 83] Dan Fass and Yorick Wilks. Preference semantics, ill-formedness, and metaphor. *American Journal of Computational Linguistics*, 9(3-4):178-187, 1983.

[Fillmore 68] C. J. Fillmore. The case for case. In E. Bach and R. Harms, editors, *Universals in Linguistic Theory*, pages 1-90, Holt, Rinehart, and Winston, New York, 1968.

[Fillmore 77] C. J. Fillmore. The case for case reopened. In P. Cole and J. M. Sadock, editors, *Syntax and Semantics VIII: Grammatical Relations*, pages 59-81, Academic Press, New York, 1977.

[Gibler83] C.D. Gibler, D.S. Childress *Adaptive dictionary for computer-based communication aids.* In RESNA 6th annual conference, pages 165-167, RESNA, San Diego, CA, 1983

[Granger 83] Richard H. Granger. The nomad system: expectation-based detection and correction of errors during understanding of syntactically and semantically ill-formed text. *American Journal of Computational Linguistics*, 9(3-4):188-196, 1983.

[Halliday 85] M. A. K. Halliday. *An Introduction to Functional Grammar.* Edward Arnold Publishers, 1985.

[Hirst 87] Hirst, Graeme. *Semantic interpretation and the resolution of ambiguity.*Cambridge University Press, Cambridge, 1987.

[Jensen et al. 83] K. Jensen, G. E. Heidorn, L. A. Miller, and Y. Ravin. Parse fitting and prose fixing: getting a hold on ill-formedness. *American Journal of Computational Linguistics*, 9(3-4):147-160, 1983.

[Kay 79] Martin Kay. Functional grammar. In *Proceedings of the 5th Annual Meeting*, Berkeley Linguistics Society, 1979.

[Kay 86] Martin Kay. Parsing in functional unification grammar. In B. Grosz, K. Sparck Jones, and B. Webber, editors, *Readings in Natural Language Processing*, pages 125-138, Morgan Kaufmann, 1986.

[Kraat 84] A. Kraat. Communication interaction between aid users and natural speakers - an international perspective. In *Proceedings of the 2nd International Conference on Rehabilitation Engineering: Special Sessions*, pages 43-46, 1984.

[Marsh & Sager 82] N. Marsh, E.and Sager. Analysis and processing of compact text. In *Proceedings of the 9th International Conference on Computational Linguistics*, pages 201-206, Coling82, July 1982.

[Marsh 83] E. Marsh. Utilizing domain-specific information for processing compact text. In *Proceedings of the Conference on Applied Natural Language Processing*, pages 99-103, ACL, 1983.

[Marsh 84] E. Marsh. A computational analysis of complex noun phrases in navy messages. In *Proceedings of the 10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association of Computational Linguistics*, pages 505-508, Coling84, Stanford University, Ca., July 1984.

[McCoy et al. 90] Kathleen F. McCoy, Patrick Demasco, Mark Jones, Christopher Pennington, Charles Rowe. *Applying Natural Language Processing Techniques to Augmentative Communication Systems*. In Proceedings of the 13th International Conference on Computational Linguistics, COLING90, Helsinki, Finland, August, 1990.

[McDonald 80] D. D. McDonald. *Natural Language Production as a Process of Decision Making Under Constraint*. PhD thesis, MIT, 1980.

[McKeown 85] K.R. McKeown. Discourse strategies for generating natural-language text.t *Artificial Intelligence*, 27(1):1-41, 1985.

[Milne 86] Robert Milne. Resolving lexical ambiguity in a deterministic parser. *Computational Linguistics Journal*, 12(1):1-12, 1986.

[Mann & Matthiessen 85] W. Mann and C. Matthiessen. Nigel: a systemic grammar for text generation. In O. Freedle, editor, *Systemic Perspectives on Discourse*, Norwood, NJ, 1985.

[Meteer et al. 87] Marie Meteer, David McDonald, Scot Anderson, David Forster, Linda Gay, Alison Huettner, and Penelope Sibun. *Mumble-86: Design and Implementation*. Technical Report COINS Tech Report 87-87a, University of Massachusetts, 1987.

[Palmer 84] Palmer, Martha. Driving Semantics for a Limited Domain. PhD Thesis, University of Edinburgh, 1984. Chapter 2: Previous Computational Approaches to Semantic Analysis.

[Rosen & Trepagnier 82] M. Rosen and C. Trepagnier. The influence of scan dimensionality on non-vocal communication rate. In *Proceedings of the 5th Annual Conference on Rehabilitation Engineering*, page 4, RESNA, 1982.

[Small & Rieger 82] Steve Small and Chuck Rieger. Parsing and comprehending with word experts (a theory and its realization). In Wendy G. Lehnert and Martin H. Ringle, editors, *Strategies for Natural Language Processing*, pages 89-147, Lawrence Erlbaum Associates, 1982.

[Swiffin 87] A. L. Swiffin, J.L. Arnott, and A. F. Newell. *The use of syntax in a predictive communication aid for the physically handicapped*. In RESNA 10th annual conference, pages 124-126, San Jose, CA, 1987

[Vanderheiden 84] Vanderheiden, G. C. A high-efficiency flexible keyboard input acceleration technique: SPEEDKEY. In *Proceedings of the Second International Conference on Rehabilitation Engineering*, pages 353-354, Washington, DC: RESNA, 1984.

[Weischedel & Sondheimer 83] Ralph M. Weischedel and Norman K. Sondheimer. Meta-rules as a basis for processing ill-formed input. *American Journal of Computational Linguistics*, 9(3-4):161-177, 1983.

[Winograd 83] Terry Winograd. *Language as a Cognitive Process*, Vol. 1: Syntax. Addison-Wesley Publishing Company, Reading MA, 1983