CERL Report X-35 October, 1972

Gregg Townsend

SPELLING, WORD, AND CONCEPT RECOGNITION

P. TENCZAR
W. GOLDEN

Computer-based Education Research Laboratory

University of Illinois Urbana Illinois

SPELLING, WORD, AND CONCEPT RECOGNITION

Paul J. Tenczar
William M. Golden
Computer-based Education Research Laboratory
University of Illinois, Urbana, Illinois

Human words must be rapidly recognized by a computer system designed to maintain an interactive dialogue with humans. We wish to discuss features of the PLATO IV - TUTOR[1] system which facilitate the matching of words and concepts offered by students to those proposed by lesson authors. We first discuss a word recognition algorithm that handles spelling errors without doing character-string manipulations or linear searches. The spelling algorithm uses human rather than machine criteria. Then, a method is described that successfully obtains the human concept from a multiple word input (e.g., a sentence) without doing tedious syntactic parsing.

The Problem

Typically a PLATO lesson author provides several answers for each question he asks students. Each answer usually consists of several words. Also, a typical student may answer several times before adequately matching any specified author answer. The computer must rapidly discover which author answer the student is attempting and then give him helpful feedback such as crossing out unrecognized words or underlining misspellings. (Figure 1 is a photograph of a PLATO IV display illustrating this feedback scheme.)

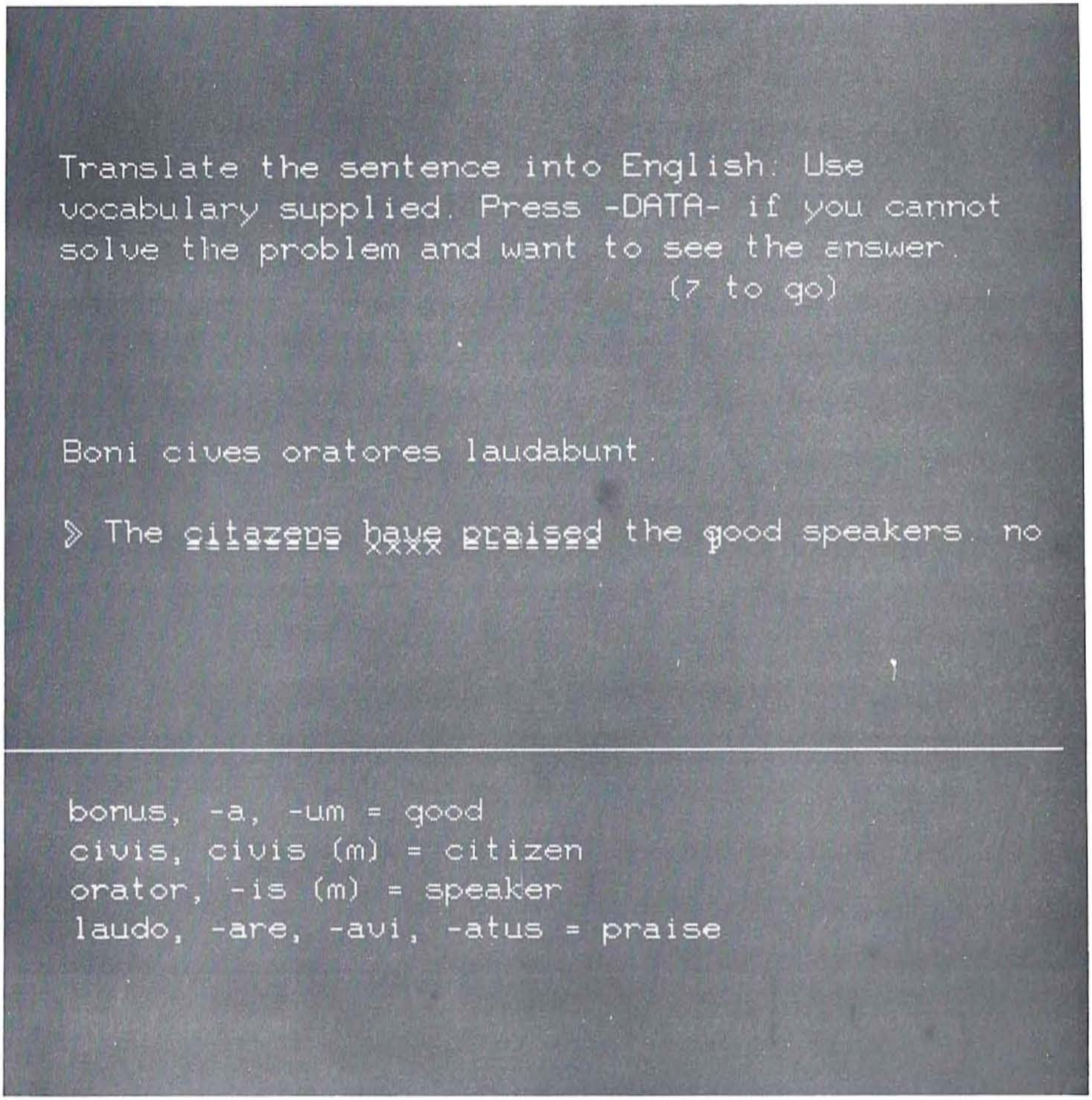
The number of procedures (each involving many computer operations) executed for a typical student can exceed:

$$\overline{N}_{total} = \overline{N}_{aa} \times \overline{N}_{aw} \times \overline{N}_{ac} \times \overline{N}_{sa} \times \overline{N}_{sw} \times \overline{N}_{sc}$$

where

$\overline{N}_{aa}$ = the expected number of author answers per question.

$\overline{N}_{aw}$ = the expected number of author words per answer. (This can be quite large if the author provides synonyms and extra words which are acceptable but not required.)



Translate the sentence into English. Use vocabulary supplied. Press -DATA- if you cannot solve the problem and want to see the answer.
(7 to go)

Boni cives oratores laudabunt.

> The citazens ~~have~~ praised the good speakers. no

bonus, -a, -um = good
civis, civis (m) = citizen
orator, -is (m) = speaker
laudo, -are, -avi, -atus = praise

FIGURE 1. This is a photograph of a PLATO IV Plasma Display Device showing a latin translation exercise from a lesson by Richard T. Scanlan, (Department of the Classics, University of Illinois, Urbana). A correct translation is, "The good citizens will praise the speakers." In the photograph, "citazens" and "praised" are marked as misspellings -- "citazens" because it is misspelled -- "praised" because it is close to the correct word "praise". The word "have" is crossed out. It doesn't belong in the translation. The word "good" should be moved toward the left. That is what the "1" indicates. (Photograph by Stanley G. Smith)

$\overline{N}_{ac}$ = the expected number of author characters per word.

$\overline{N}_{sa}$ = the expected number of student attempts at an answer.

$\overline{N}_{sw}$ = the expected number of words in a student answer.

$\overline{N}_{sc}$ = the expected number of characters per student word.

This multiplicative approach to the problem cannot be tolerated for efficient computer operation. Character-string manipulations must be essentially eliminated and word searches must be performed in a non-linear fashion -- all without loss of information! It must be remembered that useful computation begins only after the computer has recognized what the student entered. Our algorithm was designed with these considerations in mind.

When the time comes to try to recognize a student response, our algorithm does not manipulate the author's words. (Thus we eliminate $\overline{N}_{aw}$ and $\overline{N}_{ac}$ from the above product.) Also, having failed to match a student response to the first of several author-answers, subsequent attempts at matches do not re-examine the characters or words which make up the student response. Finally, that portion of the algorithm which must be repeated for each attempt at matching a word is extraordinarily short and it includes both recognition of perfect matches and of probable misspellings.

The Execution Algorithm

Both the author and student words are mapped onto a fixed number of bits. The number of bits depends on the word length of the computer used and the number of bits necessary to do the job. Thus, all "human" words regardless of their length, are mapped onto a fixed number of computer words. While the mapping algorithm is most important, its discussion will be deferred until later. For ease of discussion, we will assume a mapping onto a single computer word and call it the content word. The problem of author-student content word matching reduces to the most efficient method of discovering if:

(1) the words match perfectly;

(2) the words are possible misspellings; or

(3) the words are different.

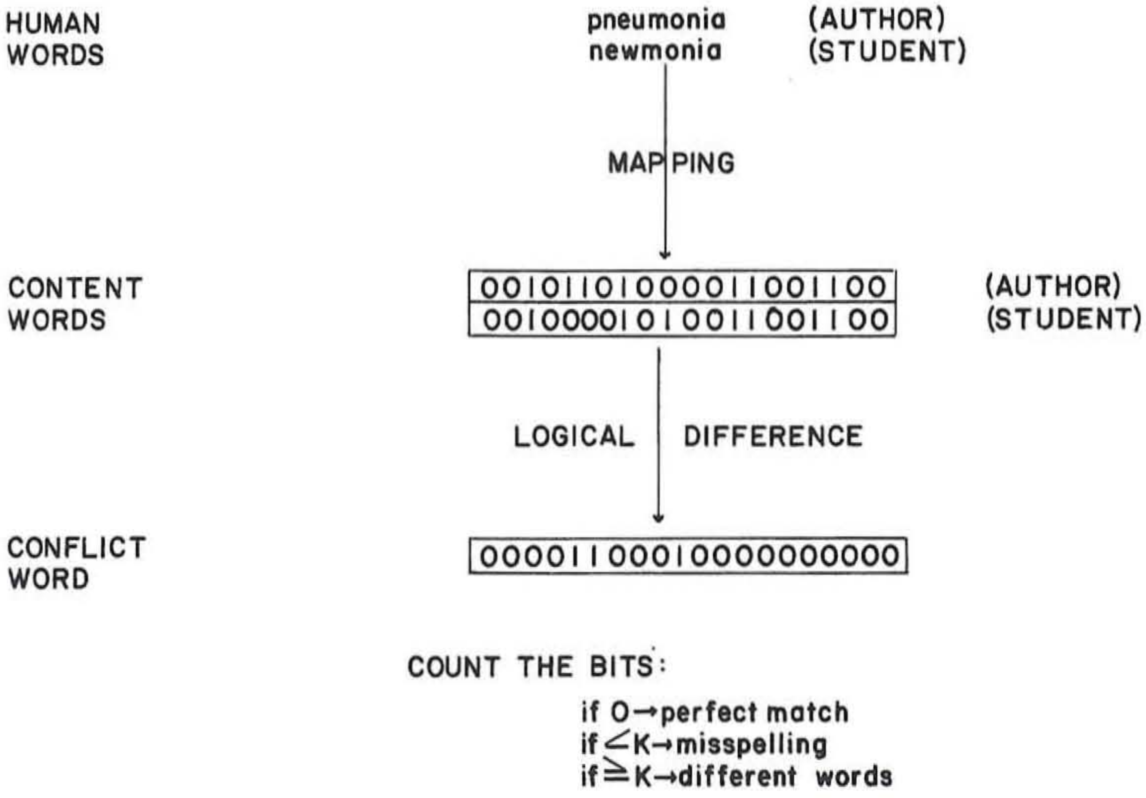
Our method involves creating a conflict word. This word has a bit pattern which is the logical difference of the author and the student content words. Upon obtaining the conflict word, it is an easy matter to count the total number of bits in conflict. We have constructed the bit setting algorithm such that:

(1) if there are zero conflict bits, then student and author used the same word;

(2) if there are fewer than a constant k conflict bits, then the student word is a possible misspelling of an author word; and

(3) if there are more than k conflict bits, then the student and author words are different.

Our procedure is shown schematically:



For the Control Data Corporation 6400 computer, these operations are encoded by these seven linear machine instructions:

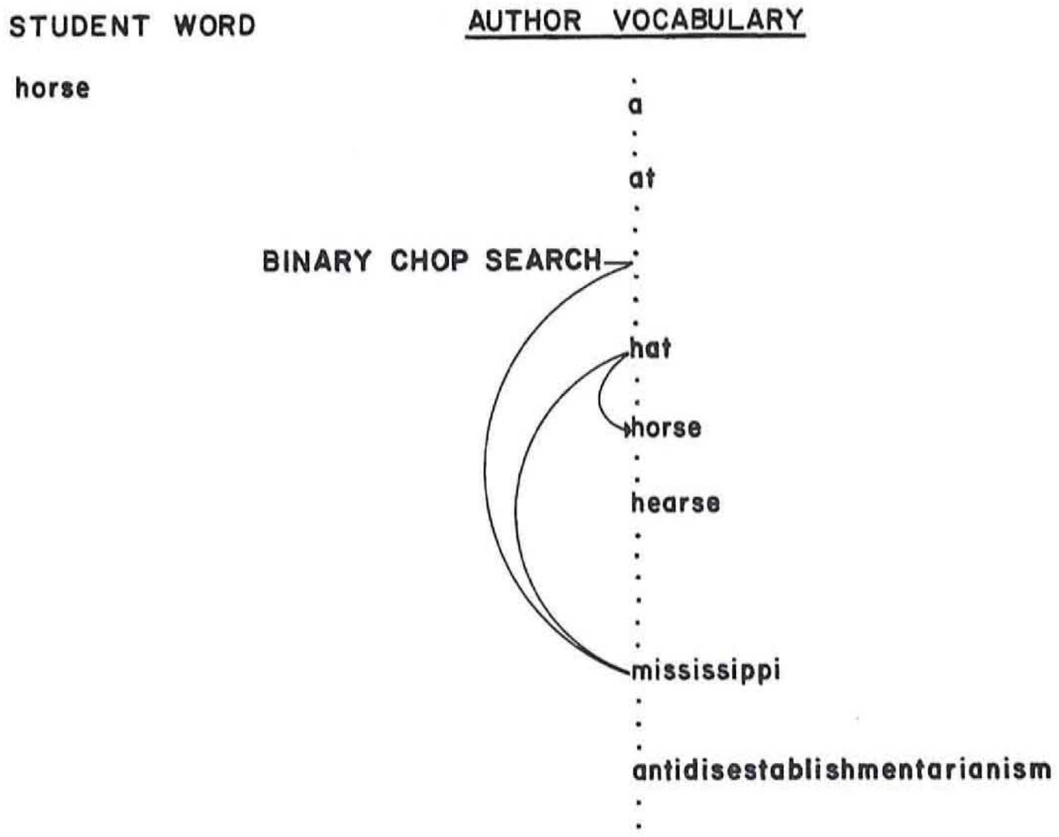
```
SA1     AUTHOR          load author word
SA2     STUDENT         load student word
BX3     X1-X2           obtain logical difference
ZR      X3, PERFECT     test for perfect match
CX3     X3              count up conflict bits
SX3     X3-K            subtract K
NG      X3, SPELL       test for possible misspellings
 .                      else different words
 .
```

While the above machine instructions do not exist in all brands of computer, simple substitutions will perform the same function. Needless to say, the elimination of character string manipulations (often called the inner loop) yields an order of magnitude speed-up in word and misspelling recognition.

Furthermore, the bit mapping algorithm is such that the resultant author words can be numerically ordered. This ordering has the result that words that look alike are listed close to each other.

```
.
.
a
.
an
.
at
.
.
.
there
.
three
.
.
yipsilanti
.
.
mississippi
.
.
```

Such a numerical ordering allows the matching algorithm to utilize a binary chop search rather than a linear search over the author vocabulary.

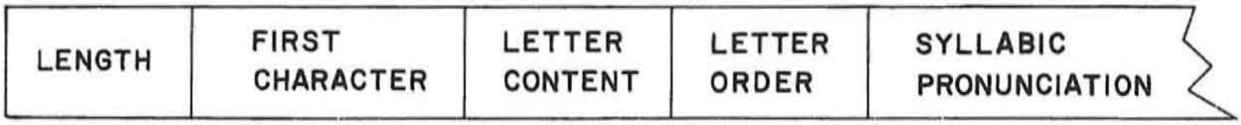


At the termination of the search, either a perfect match is found or one is left in an area of possible misspellings. The non-linearity of the search yields up to another couple orders of magnitude speed-up in computer recognition and makes the work time nearly independent of the length of the author vocabulary. This is very important at PLATO where lesson authors have generated vocabularies of up to 1,400 words within a week. Students are allowed to use any word recognized as part of the author's vocabulary.

A student word may not match any author word perfectly but may pass the spelling test for several author words. In such a case, the word producing a conflict word with the smallest number of bits set should be chosen. Note that this method performs perfect matching and misspelling matching in one efficient step. Multiple passes through the author vocabulary are never needed. Note also that the mapping of author words need occur only once; and that can be before any student begins working. Other word recognition schemes involve manipulating the author words every time any student answer needs to be recognized.

The Word Mapping Algorithm

Rather than explaining our specific algorithm in detail, we shall seek to develop a philosophy for the development of word mapping algorithms. Almost any mapping scheme will work well if it uses several human criteria to set bits. Past attempts to devise methods for recognizing spelling errors have used a minimal set of human criteria (e.g., the phonetic approach). However, no one criteria appears sufficient to do the mysterious thing which humans do when they recognize words. Rather, we should use as many features of words as we can think of and hope that the interactions between these factors will contain the information that human begins use. It is our belief that this problem illustrates a synergistic principle: the whole is greater than the sum of the parts. This technique has been successfully applied in other fields (e.g., numerical taxonomy[2]). Let us divide our content word into several subjectively selected fields. We must ask ourselves, what do we see when we look at words? Here is our current vision:



Each of these fields is assigned a bit length determined by our subjective feeling of the importance of the field. We invite others to add additional fields to ours! Each of our fields are discussed in turn.

LENGTH

The length of a word creates an obvious impression. A two letter word cannot be a misspelling of an eight letter word! As a first consideration, one might attempt to use the binary representation of the word's length in this field. The following table shows that that is undesirable:

Word length	Binary Representation	Better Representation
1	001	000
2	010	001
3	011	011
4	100	010
5	101	110
.	.	.
.	.	.
.	.	.

Using binary representation, words of one letter length difference (e.g., 3 and 4) will provide 3 conflict bits while words of many letter length differences e.g., 1 and 5 would produce only one conflict bit. A better representation would retain in the conflict bits the closeness of words of about the same length.

FIRST CHARACTER

The first letter of a word is perhaps the most dominant. We store information on this character by setting a bit to separate consonants and vowels. Then we map somewhat ambiguously, the specific consonant or vowel onto a few more bits. Although ambiguous mappings lose information, in this case the loss is only partial since the first letter will also set bits in some of the following fields. Here we see an advantage of using redundant criteria rather than a few independent measures.

LETTER CONTENT

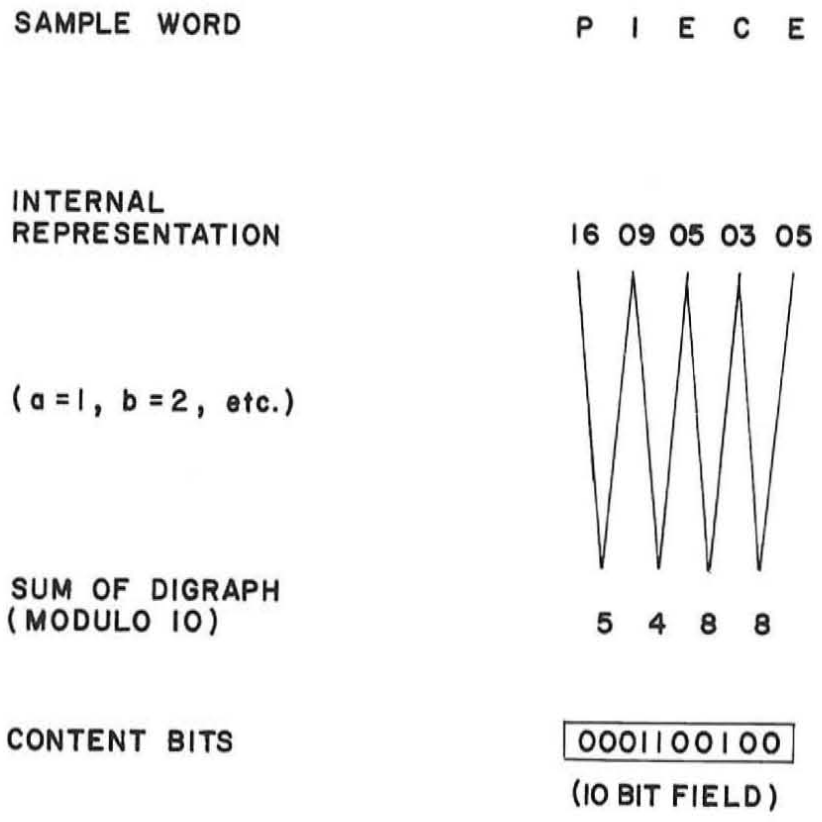
The letters of words independent of their order in the word contain information. The reader should have little difficulty in finding a word made up only of one or more occurrences of each of the letters -- s, m, p, i -- and even less difficulty in finding a word not so restricted. One need not use 26 bits for the letters of the alphabet but can map ambiguously using pairings based on phonetics and data from studies of letter usage in particular languages. For example, the word Mississippi can be mapped into the letter content field as

0	0	0	0	0	I	I	0	0	0	0	0	0	I	0	I
E	T	A	O	N	I	S	R	H	L	D	C	U	P	F	M
						Q	V	G	Y	J	K	B	W	X	Z

Here the mapping rule is partially based on letter frequencies in written English.

LETTER ORDER

We have found it useful to set bits dependent upon all the digraphs (adjacent letter pairs) present in the word. One possible scheme is illustrated:



The value of the digraph sum is used to set bits in the letter order field (the sum is done modulo the number of bits in the field). Note that the frequent spelling error of letter inversion (e.g., ei for ie) will maintain the same digraph sum and thus this information will be passed to the conflict word.

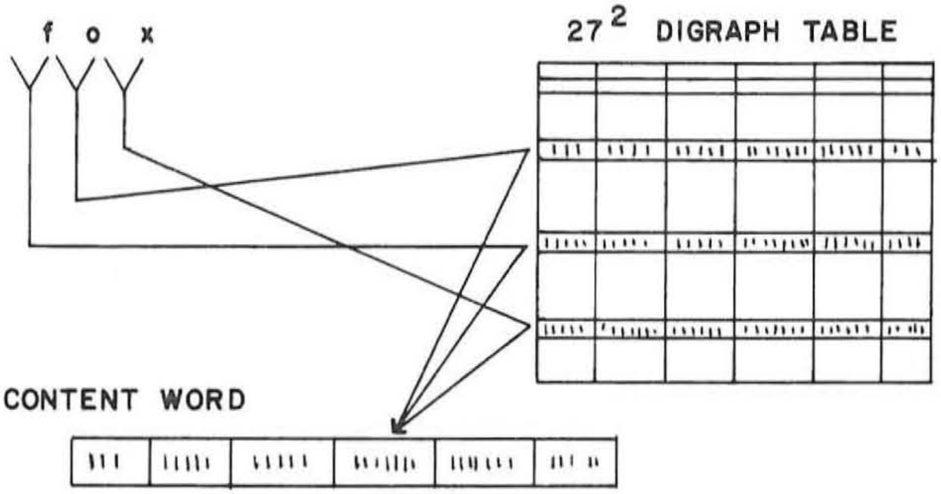
SYLLABIC PRONUNCIATION

The number and sound of syllables in a word are difficult to capture by algorithm. Our trivial but partially successful method is to use the consonant-vowel pairs in a word. Thus California becomes - ca li fo ni-. This is about the right number of syllables and pronunciation. Of course, counter-examples can be given to make the scheme look foolish. But we think this pronunciation-syllabification has enough merit to warrant a field of bits. Again, one can set these bits using the consonant-vowel digraph sum and phonetic pairing may be included where desired.

OVERALL CHARACTERISTICS

Several overall characteristics of this mapping algorithm should be noted. First, additional word properties can be easily added to the scheme without major overhaul. One need only add another bit field for each proposed new property. Second, the length and first character field at the "top" permits one to order an author's vocabulary numerically yet retain the relative closeness of similar words. A binary chop search can thus be performed on the author's vocabulary and, if no match is found, one is left in an area in which misspelling matches can occur. Third, the exact bit coding schemes and the field lengths are easily varied so one can heed the information theory dictum which states that in a coding scheme one should strive to have a probability of 0.5 of finding any given bit set.

Finally we must dispel the idea that this word mapping algorithm is tedious. It is not since all fields but the length field can be set simultaneously doing only N table look ups where N is the number of characters in the word. This is done by doing a table look up for each digraph in the word and then "or-ing" these table entries together. (One uses a "space-first letter" pair in the first table look up to incorporate the first letter field information.) This is shown schematically:



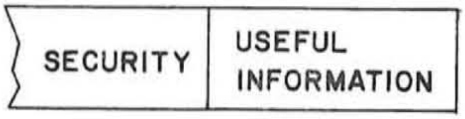
We have discovered that more information about words is retained if some fields of the content word are set by an exclusive or operation instead of an or operation.

In leaving this section, we note once again that for words of the author's vocabulary the mapping process occurs only once during compilation time. This yields a considerable run-time efficiency since only student input need be "contented" during execution.

NUMBERS

Authors most often prefer that numerical parts of student responses be treated numerically rather than as words. Thus the system should recognize that 106 nearly matches the expected value 105.84 and that 3/4 does match the expected value 0.75. On the other hand, for sake of efficiency, numbers must fit into the same scheme used for recognizing words. They do. We compile numbers (including preceding + and - signs) into floating point representations. These floating point numbers are then stored in content words. The top bit of every content word is reserved for distinguishing words from numbers. Words have that bit set 0 while numbers have the bit set 1. Content words formed from numbers, for some purposes, can be handled like those from words. Words and numbers can be interspersed in a vocabulary -- they will be separated when the vocabulary is numerically sorted during compilation. A binary chop search can be used to find perfect matches or the area of possible misspellings. (For numbers that means small differences.) If no perfect match is found and we are in a "negative" vocabulary area, then we unpack the floating point portion of the content word and test for small differences between the author and student numbers. Generally we accept a 1% difference and treat differences of more than 1% but less than 10% in a manner similar to misspellings. These criteria can be readily altered by the author. Finally, in compiling a number we continue through any operator and the following number. Thus the answer 3/4 of the pie becomes the same as 0.75 of the pie. This option can be turned off by an author.

ADDITIONAL FIELDS



The security field contains a hash representation of the word. The field is large enough to make negligible the probability that two different words will accidentally map onto the same hash bit pattern. Obviously, this field should be masked out of the conflict bits before they are counted. Finally, a field containing syntactic, semantic, or other useful information concerning the word may be included. A use of this last field will be discussed later in this paper.

Tests of the Spelling Algorithm

Several coding schemes have been tried. One which performs satisfactorily uses 41 bits to record information about the length of the word; the first character, the consonants, the syllabification, and the pairwise order of characters. The criterion for saying that a student word is a misspelling of the author word is quite simple -- fewer than 7 conflict bits!

We have several procedures by which such coding schemes are tested. A dictionary of common misspellings in English[3] is used to see whether a proposed algorithm will recognize those misspellings which people in fact produce. The 41-bit algorithm recognizes over 95% of a sample of items taken from this dictionary. For example, the algorithm recognized the following misspellings -- chevrolay for chevrolet, angziety for anxiety, fantom for phantom. To test that the algorithm is not too lenient, pairs of successive words from an ordinary dictionary are proposed as misspellings of each other. Such pairs of words tend to be quite close in spelling. Fewer than 14% of a sample of these pairs were mistakenly identified as misspellings. Another test generates 50,000 random letter sequences which obey the letter frequency counts determined for English.

These letter sequences also randomly vary in length from 1 to 15 letters. Each of these is tested against every one of the others and the pairs which the algorithm calls misspellings are printed. Ideally, each pair printed should look like a misspelling to a human observer. In repeated trials, fewer than 50 of these 2.5 billion test pairs are considered misspellings by the algorithm but not by human observers. No accidental exact match ever occurred.

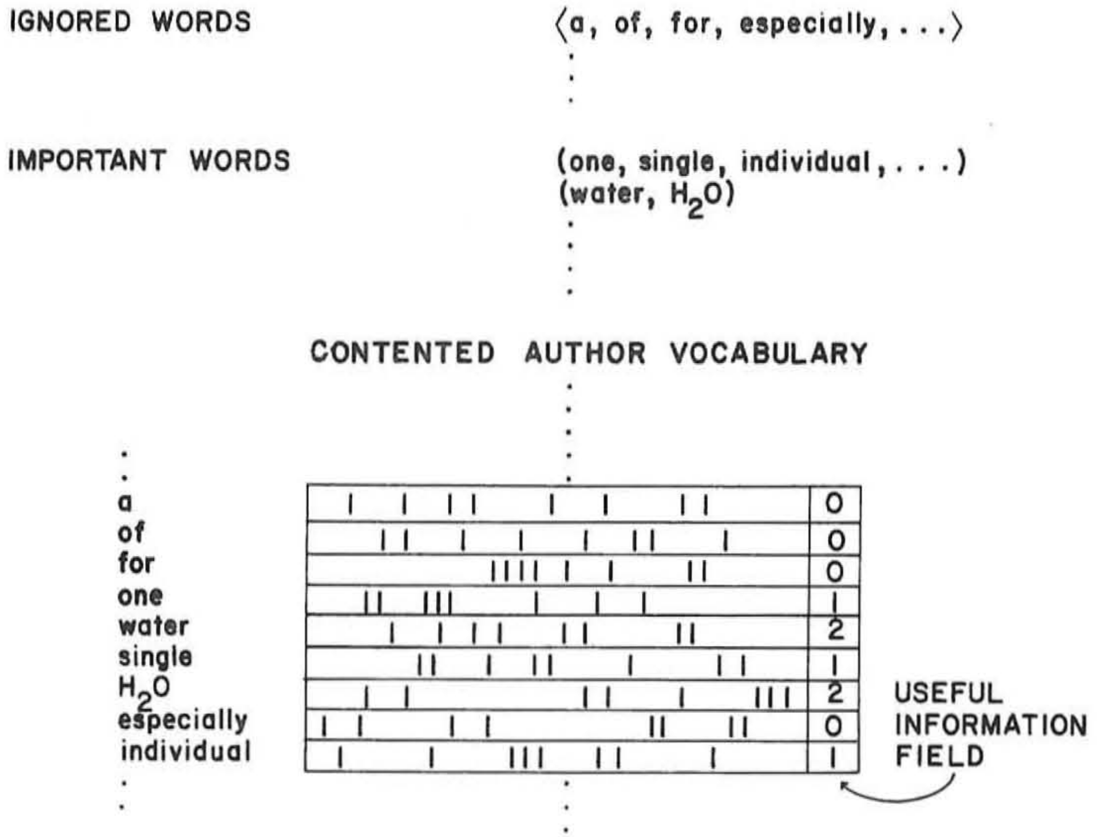
The most critical test we have found consists of checking each of the 500 most commonly used English words against each other. A good algorithm must separate all but the closest of these words. Despite the fact that this list contains no words over 6 characters in length, our algorithm performed satisfactorily, (i.e., pairs which it calls misspellings usually differ by only one letter).

As an outgrowth of this last test we have come to the conclusion that, for students proficient in English, if a student word matches perfectly any of the 500 most common English words, then we ought to assume that that is the word the student meant. If an educated person types the word "food" and this word does not appear in the author's vocabulary, it is safer to say the student has an incorrect word than a misspelling of "foot". Since a binary chop search through the common vocabulary list is so fast (about 50 microseconds for a CDC 6400) it seems desirable to perform such a search before declaring a misspelling to the student.

Concept Recognition

Consider a computer system that accepts freely-worded student discourse in a subject area. For example, a medical student caring for a hypothetical patient might ask: "What are the results of a blood serum analysis?" or "How old is the patient?" The computer must quickly "understand" these phrases and make appropriate replies. This section describes the simple but successful algorithms used to perform this concept recognition task.

Words possible in student phrases are entered into a lesson vocabulary by the author. These words are identified as ignorable words or important words. Important words are organized into lists of synonymous words. Both knowledge of the subject area and a thesaurus are useful in establishing synonym lists. An example of a list of synonymous important words is -- one, 1, single, alone, only, unit, individual. Examples of ignored words might be -- the, what, did, a, with, of. Ignored words are given the value 0 in the useful information field of their content words. Important words are assigned the present value of a counter of important words in the useful information field of their content words. All synonymous words are assigned the same value. This is illustrated below. [<...> denote ignorable words while (...) denote sets of synonymous words.]



A phrase consists of a list of words present in the vocabulary (e.g., "give me the differential white blood cell count"). Ignorable words (identified by 0's) are cleaned out of both the author's concept and the student's response.

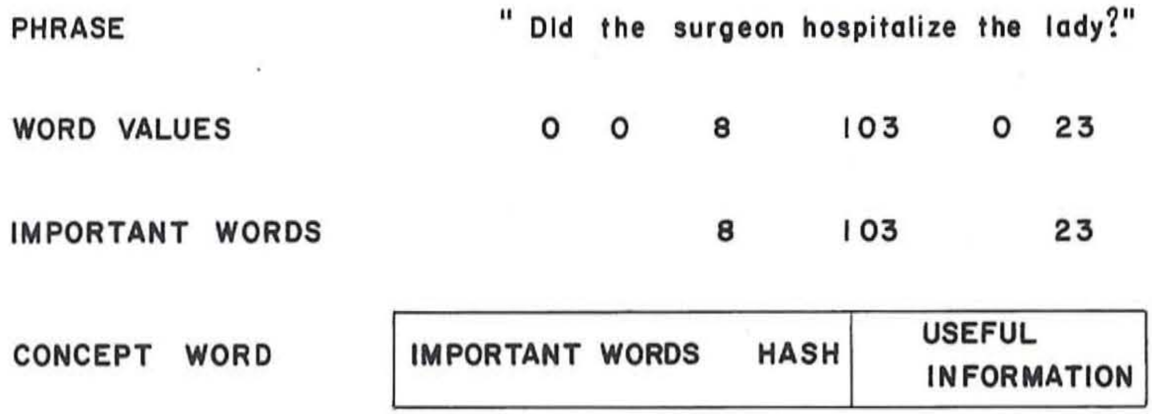
Great power can be achieved by the author through correct synonym construction. For example, consider the following synonym list:

(doctor, doc, surgeon, physician, pathologist)
(patient, woman, female, lady, girl)
(visit, see, consult, interview, call)
(lately, recently, late, yesterday)

The author enters only the phrase -- did the patient visit the doctor recently? -- and, with interspersed ignored words, the computer can "understand" over 500 possible responses. Of course, care must be taken to avoid joining two different human concepts into one computer phrase because of faulty synonymy. Conversely, loose synonymy is not a severe hazard to concept building since the other words in the phrase will almost certainly clarify any single word ambiguities.

The Concept Word

We construct a concept word which consists of a hash-coding of all the important word values of a phrase plus a field of useful information.



PHRASE	" Did	the	surgeon	hospitalize	the	lady?"
WORD VALUES	0	0	8	103	0	23
IMPORTANT WORDS			8	103		23

CONCEPT WORD	IMPORTANT WORDS HASH	USEFUL INFORMATION

The order of important words is maintained in the computer's hash -- a fact which imparts much syntactic information upon the concept word. However, if desired, the order of important words in a phrase can be ignored simply by numerically ordering the important word values before the hash-coding operation.

Usually, at PLATO, concept recognition is used in a situation in which the student may respond with any of numerous possible concepts. The medical diagnosis exercise already alluded to is a good example, for there the student may request any one of several hundred possible pieces of information. The computer must quickly recognize which of these many concepts the student is referring to. Our algorithm operates as follows:

First, each author concept is given a value in the useful information field of its concept word. Synonymous concepts are given the same value. For example, "did the patient see a doctor recently?" and "was medical attention given to the woman lately?" refer to the same concept for the given context. Note that in this case, synonymous words are not enough to make the two phrases map onto one concept.

Second, all of the concepts for a given response situation are numerically ordered. This allows a binary chop search for a student-author concept match and makes irrelevant the number of author concepts. Thus, when a student enters a phrase (1) each word of the phrase is first "contented" then matched to a word in the author vocabulary; (2) the values stored in the useful information fields of those author words are noted; (3) using those values, the student's phrase is converted into a concept word; (4) this concept word is matched by binary-chop search to an author concept; and (5) the useful information field of the matched author concept is then available to provide a proper computer response to the student. Using the CDC 6400 the entire process to match a student's ten-word phrase to one of several hundred author concepts using a thousand-word vocabulary takes between 1 and 2 milliseconds.

Expressed in the same variables as were used in the formula which appeared earlier, the number of procedures executed for a typical student is:

$$\boxed{\bar{N}_{total} = \underset{④}{\bar{N}_{sa}} \left(\underbrace{\log_2 \bar{N}_{aa}}_{③} + \underbrace{\bar{N}_{sw} \times \log_2 \bar{N}_{aw}}_{②} + \underbrace{\bar{N}_{sw} \times \bar{N}_{sc}}_{①} \right)}$$

and the interpretation of these terms is:

① Examine each student character and word <u>once</u> to produce a content word;

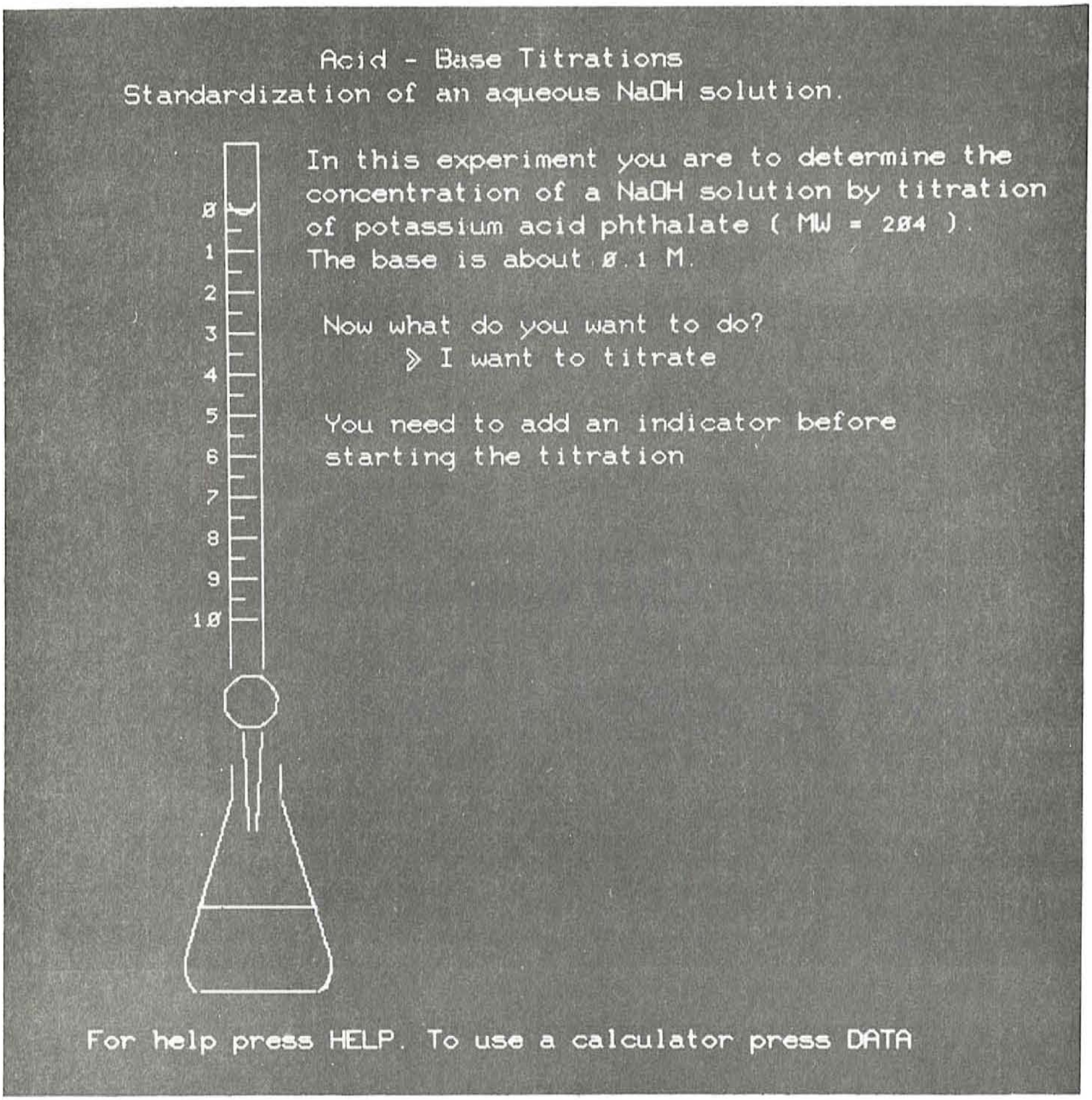
② Find each student word in the author's vocabulary via binary-chop techniques;

③ Perform a binary-chop search through the author's concepts;

④ Do all of the above for each student answer attempt.

Before going on to examples of the application of concept recognition, some minor details should be considered. Generally at PLATO, the fact that words are capitalized is ignored. This greatly eases the difficulty of vocabulary construction. However, for many subjects this would prove a disaster and then it is not done (e.g., in chemistry, h_2so_4 is not acceptable).

Again, generally at PLATO, student words that do not exist in the author's vocabulary immediately abort the concept building processes. These words are crossed out of the student's response and a message is given to the student informing him that the indicated words are not part of the lesson's vocabulary. For the case in which all words are recognized but no author concept is matched, most authors return to the student a message like, "Sorry, but I do not understand. Try rephrasing."



Acid - Base Titrations

Standardization of an aqueous NaOH solution.

In this experiment you are to determine the concentration of a NaOH solution by titration of potassium acid phthalate (MW = 2ø4). The base is about ø.1 M.

Now what do you want to do?

> I want to titrate

You need to add an indicator before starting the titration

For help press HELP. To use a calculator press DATA

FIGURE 2. Photograph of a PLATO IV Plasma Display Device showing a portion of the lesson ACID2 by Stanley G. Smith (Department of Chemistry, University of Illinois, Urbana). This student is 3 or 4 steps along the way toward performing an acid-base titration. Concept recognition is used to understand the student's requests. Before the lesson is over, the student will have requested all the necessary steps in the right order, "operated" the equipment with sufficient care, and performed the correct calculations to determine the concentration of NaOH. (Photograph by Stanley G. Smith.)

Applications of Concept Recognition

For some lessons it is pedagogically advisable to ask open-ended questions which require students to answer in their own words. An example is a lesson on the rights of the accused in which we ask each student (who plays the role of the accused) to tell his attorney (played by PLATO) about his interactions with the police.[4] The relevance (to civil rights matters) of each of a student's statements is commented upon, and when a student indicates he has nothing more to say, he is reminded of any infringements of his rights which he forgot to mention.

Another use of concept recognition permits a reversal of roles. Students ask questions; the computer answers. Such lessons fit what might be called the "trouble-shooting skills pattern." Medical diagnosis is such a skill. So is trouble-shooting of any mechanical or electronic device. Even qualitative or quantitative analyses in chemistry fit the pattern [see example below]. Some games (e.g. "20 questions") share this structure. What is common to these situations is that the computer knows the state of some system and students ask questions to discover that state. Answers may be stored but they are more likely to be the results of calculations. We have examples of this second type of lesson in the fields of chemistry, French phonetics, mathematics, philosophy of science, and veterinary medicine. In the absence of concept recognition, such lessons might resort to having students choose the category of information desired from a list of categories. Any such procedure is far less like the clinical situation for which students are being trained than is a lesson which effectively uses concept recognition.

A third use involves students in specifying a very complex procedure which consists of many steps, and for which there are many acceptable orderings of the steps. (Certain steps must be performed at some time prior to certain others, and some steps become optional in the presence of others.) (Figure 2 is a photograph of a PLATO IV display generated by a chemistry lesson of this type.) Such lessons are especially effective if they include a simulation of the process which produces graphical or other output to show the effect of each step as it is suggested by a student. We have several examples from science education. In one chemistry lesson, students are asked to purify naptholene by recrystallization from a methanol solution.[5] Each student specifies the steps in his own words without hints from context. Sometimes errors are explained as they are made and the erroneous steps are rejected. Other errors evoke no remark and the student must live with the consequences of his actions. Before a student is said to have completed the lesson, he has specified the purification process from start to finish with at most trivial errors. A similar biology lesson requires specification of the cell parts (including DNA and RNA codons) needed to make a given short protein chain.[6] In this case, student inputs tend to be short enough to be handled by other recognition techniques. Concept recognition was chosen because of the ease of authoring using that technique.

Following are excerpts from a PLATO lesson written in TUTOR which gives college students practice in the intellectual aspects of organic qualitative analysis.[7] Shown are samples of the coding necessary for recognition of student questions and the answering of them. The structure of the lesson is simple. The computer randomly chooses an unknown as each student begins. Students ask questions in an attempt to identify the unknown. A score is kept which measures the efficiency of each analytical scheme. In its last use, with 14 students, this program recognized over 90% of the questions asked. [The lesson is improved after (or even during) each use.] This level of success is typical of the experience PLATO authors have had using concept recognition. For most questions which are not recognized, the program marks the words which are causing difficulty (i.e., they are misspelled or they are missing from the vocabulary) thus aiding the student in rephrasing his question.

Excerpt A

VOCAB CHEM
<a, at, an, and, add, addition,
affect, ... where, what's, you>

The lesson begins with a VOCAB command which signals the start of a vocabulary. The vocabulary is named for future reference. [A lesson may use more than one vocabulary.] The first set of words is surrounded by '<' and '>' indicating that these are acceptable words but ignorable.

Excerpt B

physical property
(odor, smell, fragrance, odorless,
fragrant...scent, stink)
(density, dense, gravity)
(soluble, solubility, dissolve, mix,
solvent)
(water, aqueous, H_2O)
(methanol, MeOH, CH_3OH)

.

.

.

(Br_2, bromination, brominate)

.

(heat, reflex, reflexing)

.

Here are the important words. Words enclosed within parentheses are synonyms. [Note: content of the lesson dictates synonomy. Often words which are otherwise unrelated should be considered synonyms.] In all, this vocabulary contains 825 words. [One of our lessons on veterinary medical diagnosis has a 1400 word vocabulary. It answers 84 distinct questions about each clinical case.[8]]

Excerpt C

CONCEPT odor
CONCEPT refraction
.
.
CONCEPT taste
CONCEPT boiling point
CONCEPT melting point
.
.
CONCEPT CrO_3
CrO_3 oxidation
chromic acid oxidation
Jones oxidation
.
.
CONCEPT nitrate
nitric acid
HNO_3 and H_2SO_4
CONCEPT react with thionyl chloride
and then NH_3
make acid chloride and then
treat with NH_3

Each CONCEPT command indicates an idea (question) the student might present (ask). If there are several entries after the command, (see chromic acid example), these are synonymous phrases which convey the same concept. The sample lesson includes 96 CONCEPT commands i.e. 96 distinct questions which can be answered. Each question, of course, might be phrased in a great many ways using the vocabulary provided.

Excerpt D

```
WRITEC   ANSCNT,
         it has a strong aromatic odor,
         the refractive index is 1.5764.,
         the density is 1.119, ...
         tasting the unknowns is not
            advised, ...
         the mass spectrum is not
            available -- too costly
         there is no reaction with AgNO3 ...
```

When a student input is recognized as one of the above concepts, a corresponding number is placed in the variable called ANSCNT. WRITEC is a conditional WRITE statement which outputs one of the following messages depending upon the value of ANSCNT. Thus the student's question is answered. [Alternatively, one could branch on ANSCNT to coding which gives a full screen response, calculates a response to the student input, selects a photographic slide, etc.]

In order to add a new unknown, the author need only add a new set of responses, perhaps one or two new concepts, and a few new vocabulary items which didn't pertain to the earlier unknowns.

Computer-assisted Authoring

Lessons which use concept recognition on PLATO III offer several automated aids to authoring. Authors construct and add to their lesson vocabularies in an interactive manner. After entering a word, the author is asked to classify it as an ignorable word, a new important word, or a synonym of an important word already in the vocabulary. The computer checks that the proposed word is not already in the vocabulary and that the proposed synonym is, and then either adds the word to the vocabulary or rejects the word with appropriate comments. Even a very large vocabulary can be quickly entered. A similar on-line feature permits the addition of new concepts with their corresponding responses.

Once a basic set of vocabulary, concepts, and responses have been specified, the entire lesson can be improved through an on-line monitoring feature. An author at a PLATO III terminal identifies himself. Two or three students attempt to study the lesson at other terminals. Every time a student enters an unrecognized word, that word appears on the author's screen. He can reject the word or enter it into the vocabulary as an ignorable word, a new word, or a synonym. His decision dynamically changes the lesson for all future users. When the computer fails to recognize a phrase entered by a student, (but does recognize each of the words), that phrase is displayed to the author. He selects the correct response and that response is sent to the initiating student. (Students normally wait a tenth of a second for a computer-generated response. During this dynamic editing the student might wait 10 or 15 seconds.) The phrase, or rather the concept which it expresses, also becomes part of the lesson and will automatically trigger the proper response the next time a student enters that same concept.

When a lesson is at an early stage of development, an author is kept busy monitoring 2 or 3 students. Soon, an entire class can work in the lesson simultaneously and the author has little monitoring to do.

Acknowledgment:

The senior author wishes to express his gratitude to W. Ross Ashby for showing him the methods of "doing the job right."

[1] For a general discussion of the PLATO system and its uses see:

(a) Alpert, D. and Bitzer, D. L., "Advances in Computer-based Education", Science, 167, 1582, (1970).

(b) Bitzer, D. L. and Johnson, R. L., "PLATO: A Computer-based System Used in the Engineering of Education", Proceedings of the Institute of Electrical and Electronic Engineers, 59, 960, (1971).

(c) Bitzer, D. L.; Blomme, R. W.; Sherwood, B. A.; and Tenczar, P., "The PLATO System and Science Education", presented at the Conference on Computers in Undergraduate Science Education, Illinois Institute of Technology, August, 1970; Conference proceeding available from American Institute of Physics, 335 East 45th Street, New York, New York, pages 335-378.

[2] Sokal, R. R. and Sneath, P. H. A. Principles of Numerical Taxonomy. W. H. Freeman and Co., San Francisco and London, 1963, 359 pp.

[3] Jordan, J. (Ed.), A Handbook for Terrible Spellers. Inovation Press, New York, 1963.

[4] We refer to the lesson entitled ARREST by Myrna A. Golden, Urbana, Illinois.

[5] We refer to the lesson CRY2 by Stanley G. Smith, Department of Chemistry, University of Illinois, Urbana, Illinois.

[6] We refer to the lesson CELL by Paul Tenczar, Computer-based Education Research Laboratory, University of Illinois, Urbana, Illinois.

[7] The sample is taken from the lesson QUAL by Stanley G. Smith, Department of Chemistry, University of Illinois, Urbana, Illinois.

[8] We refer to the lesson DOG by George Grimes, School of Veterinary Medicine, University of Illinois, Urbana, Illinois.