Question Answering from Natural Language Medical Data Bases

Ralph Grishman and Lynette Hirschman

New York University, NY 10012, U.S.A.

Recommended by N. S. Sridharan

ABSTRACT

This paper describes a system for automatically answering questions about a collection of natural language medical records. The particular records used for an initial experiment were a set of 206 radiology reports. The implementation involves two major steps: manual determination of a suitable tabular structure (information format) *for representing the information contained in the medical records, and automatic conversion of the natural language input (for either record or question) into a form corresponding to the data base. For the medical records the conversion into a data base is done by first performing a syntactic and transformational analysis of the sentences, followed by application of* formatting transformations. *The question-answering procedure has analogous initial steps but undergoes additional steps of processing to translate the question into a retrieval operation on the data base. Samples of the data base and of the question-answering procedure are shown.*

Introduction

As the amount of information in computer data bases has grown, so has interest in methods to aid the user in retrieving information from data bases. One approach which has been pursued is the development of question–answering systems, through which a user could obtain information by conversing with the system in natural language. Such systems have been under development for quite a few years (Simmons [1], Petrick [2]); among the best known are Woods' LSNLIS system [3] and Petrick and Plath's REQUEST system [4]. Many such systems are currently under construction (Waltz [5]).

These systems operate on information stored in tables or other simple data structures. Much of the information needed by scientists, physicians, and technical personnel, however, resides not in tables but in textual material—in books, journal articles, reports, medical records. The Linguistic String Project of New York University is engaged in a long term effort to develop techniques for processing such material, so that it will be possible to gather statistics and answer questions automatically from natural language data bases. This paper will describe a part of that effort, a question–answering system which is able to extract information from a file of natural language documents. The system now operates on data bases of

Artificial Intelligence **11** (1978), 25–43

medical records written in unrestricted natural language. The initial experiments described here operated on a collection of 206 radiology reports for 13 patients.[1]

Medical Records as a Data Base

In order to deal with the constantly growing volume and detail of information that health care practitioners are required to provide (for patient care evaluation, public health statistics, clinical research, etc.) many hospitals have introduced computerized hospital records systems. Although this is an important first step, it does not permit automatic encoding and retrieval of the information in the natural language records. There are currently a number of experimental systems that provide for the automatic encoding of a medical record. One type is based on a checklist, filled out by the doctor to generate the record (Morgan [6]). This approach has the disadvantage that it constrains the type of information that can be reported, and turns the health care worker into a high-class coder. Another approach (Dunham et al. [7]) uses a minimal syntactic analysis and an extensive nomenclature (such as SNOP or SNOMED) to encode medical records. Although this type of processing has been successful for certain kinds of medical information (e.g., pathology reports, which consist almost entirely of noun phrases), it cannot handle syntactically complex information, in particular coordinate conjunction and scope of negation. In contrast to these experiments, the approach of the Linguistic String Project is based on syntactic processing. This allows the health care worker to create the medical record in the most natural way—in medical English, using whatever syntax is appropriate to the information. The natural language records are then automatically processed to produce a data base.

Our ability to process this information is due to the fact that the language used in these documents constitutes a *sublanguage*: a specialized subset of English dealing with circumscribed subject matter. In particular for medical records, the vocabulary, while not small, is extremely limited compared with English as a whole. Syntactic constructions are less rich than in other types of discourse. While the frequent deletion of certain verbs and the ubiquitous occurrences of run-on sentences poses some difficulties for processing, on the whole the material is straightforward fact-reporting prose, with a high degree of specialized usage (few homonyms) and a minimum of ambiguity.

The limited subject matter of medical records also contributes to its computability. Only certain topics are dealt with: for the most part, observations of the patient's state and descriptions of tests, test results, and treatments. All of these factors combined to make these texts suitable candidates for the text-structuring procedures we shall describe.

Formatting

The heart of our system is the transformation of the text into a tabular structure called an *information format*. Each row of the table will correspond to a sentence

[1] This material was furnished to us by Irwin D. J. Bross of Roswell Park Memorial Institute.

or part of a sentence in the text. Words of different sentences which play the same informational role are placed in the same format column. Fig. 1 shows a greatly simplified format for X-ray reports, along with the format entries for three sentences.

Because words conveying information of a particular type are aligned under a single column of the format, it is much easier to write question-answering procedures which work from a formatted data base than from a textual data base. For example, the body part associated with a particular test can be immediately located in the TESTLOC column in Fig. 1. At the same time, no information is lost in translating a sentence into a format entry—the original sentence is reconstructable (up to sublanguage paraphrase) from the format entry. Consequently, any question which is answerable from the text should be answerable from the format entries.

	TEST			VERB	FINDING
	TESTN	TESTLOC	DATE		
Br 0.1.3	film	chest			negative
Br 6.1.5	x-rays		11–29–65		multiple pulmonary metastasis
Br 11.1.12	films	of chest	5–25–65	show	no callus formation
	films	of right shoulder	5–25–65	show	no callus formation

Original sentences:

Br 0.1.3 Chest film negative.
Br 6.1.5 X-rays 11–29–65 multiple pulmonary metastasis.
Br 11.1.12 5–25–65 films of chest and right shoulder show no callus formation.

FIG. 1. Simplified format.

Our system is thus divided into two components—a formatting procedure which creates the formatted data base and a retrieval or question-answering procedure which extracts information from the data base (Fig. 2). The first step in the generation of such a system for a class of documents is to define the format; we shall

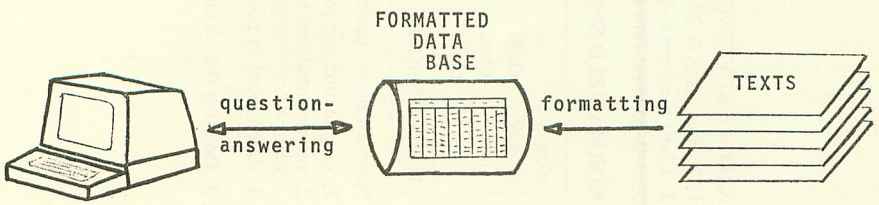

FIG. 2. Basic structure of the question-answering system.

SAMPLE SENTENCES FROM RADIOLOGY REPORTS

1. X-rays taken 3–22–65 reveal no evidence of metastatic disease.
2. Chest x-ray on 8–12–69 showed no metastatic disease.
3. 3–2–65 chest film shows clouding along left thorax and pleural thickening.

NORMALIZED SENTENCES WITH WORD CLASSES ALIGNED; bottom line shows word classes

1.	x-rays		taken		3–22–65	reveal	no	evidence	of	metastatic disease			
2.	x-ray	chest		on	8–12–69	showed	no			metastatic disease			
3.	film	chest			3–2–65	shows				clouding	along	left	thorax
	film	chest			3–2–65	shows				thickening			pleural
	NTEST	NBODY	VDONE	P	DATE	VSHOW	NEG	NSHOW	P	NCONDITION	P	ADJLOC	BODYPT

Normalizations performed:

1. conjunction expanded into two conjoined assertions (example 3)
2. pre-nominal nouns and adjectives derived from nouns moved to post-nominal position: *chest x-ray*=*x-ray* (*of*) *chest* (examples 2 and 3).
3. date placed in post subject position (example 3).

FIG. 3. Defining the format.

briefly describe a procedure for doing so in the next section. Second, formatting procedures must be written to map sentences into the defined format; these will be described in the following section. Finally, retrieval or question-answering procedures must be prepared. For statistical summaries or record evaluation according to fixed criteria, hand-coded retrieval programs are used. For on-line retrieval of specific information, a question-answering procedure can translate the English query into the appropriate data base retrieval request. Examples of each will be given in the later sections of this paper.

Defining the Format

In order to convert natural language texts of a subfield into a data base we must determine an appropriate structure for the data base. We want to convert the input text into a data base without loss of information; we also want the data base to be general in form, specialized to the subfield but not tailored to a specific use of the data. We arrive at the appropriate data base structure through a procedure based on an analysis of the linguistic regularities found in the texts of a subfield. The format obtained in this way is then verified by an expert in the field. Because construction of the format relies primarily on linguistic analysis, rather than on detailed knowledge of the subfield, it is a technique that can be applied to different subfields; it has been successfully applied to pharmacology articles (Sager [8]), the X-ray reports described here (Hirschman et al. [9]), and currently to hospital discharge summaries (Sager and Lyman [10]).

The data base is structured as a table or *information format*, whose columns correspond to the types of information found in the texts of the subfield. The entries of the columns in a given row are the actual words of the input sentence. Definition of the format consists of two steps: identifying the elementary informational units (word classes) of the sublanguage, and determining how these units are related in the sentences of a sublanguage text.

Word classes are formed by grouping together words occurring in similar syntactic environments. For example, in the sentences of Fig. 3, we group together *X-ray(s)* and *film* which both occur as the subject of the verb *show*; similarly *reveal* and *show* are grouped into a class because they both occur with the subject *X-ray*. Because of the relationship between the distribution of a word and its meaning, the words occurring in similar environments carry related information (e.g., *X-ray*, *film*). In fact these word classes are the informational units that are needed to structure the data base.

The word classes for the radiology material were prepared manually. However, we have developed a program that generates automatically the major word classes of a sublanguage from a sample set of syntactically normalized (transformationally decomposed) sentences. Each normalized sentence is processed to obtain a set of triples, consisting of the name of a syntactic relation and the two words in that relation. For example the sentence *lateral x-ray shows lesion* yields the triples

verb-subject	show	x-ray
verb-object	show	lesion
host-adjunct	x-ray	lateral

Each word in the corpus is then associated with a vector which represents its occurrences with each other word in a particular syntactic relation. A similarity coefficient for each pair of words is computed as the inner product of their vectors. Two words have a nonzero similarity coefficient only if they have both occurred with the same word in a particular syntactic relation. Words are then grouped into clusters if their average similarity coefficient exceeds a threshold value. This method was tested on a manually normalized corpus of pharmacology texts; it generated the major word classes in good agreement with the manually determined classes (Hirschman et al. [11]). This program is currently being tested on a set of automatically normalized sentences from the radiology corpus.

Next we normalize the sentences using English paraphrastic transformations and align identical word classes in the various sentences, as shown in Fig. 3. (The word classes are listed below the three aligned sentences; where a word does not have a sublanguage word class, its English word class is used, e.g., *P*(*reposition*), *NEG*(*ation*).) This alignment creates a sequence of word classes which is gradually refined into a format for the sublanguage. On the one hand, columns are added as required by further sentences in the sample. On the other hand, some of the columns shown in Fig. 3 can be eliminated. Format columns are only needed for word classes that carry independent sublanguage information. Therefore we do not need the column P for the preposition *of* in *no evidence of metastatic disease*, since *of* here is simply a syntactic marker. We also do not create a separate column for *left* in *clouding along left thorax*; instead we leave *left* as an adjunct on thorax, treating the expression as a unit. Fig. 4 shows an abridged version of the final format, and five formatted sentences.

This format has the desired properties: we can read off the sentence (up to paraphrase) from the column entries of its formatted version, indicating that no information has been lost. The relevant informational units correspond to the format columns and we can locate each specific type of information by looking at the entry in the appropriate column (e.g., kind of test performed is found in TESTN column). Finally because the format corresponds to the linguistic structure of the original sentences, it is possible to define a set of "formatting transformations" that map each sentence into the appropriate format columns.

Formatting the Sentence

Once an appropriate tabular structure has been established using the (manual) procedure described in the preceding section, the next step is to transform the input sentences automatically into entries in this table. For this we need two kinds of information: word class membership and syntactic relations between the words.

Input text sentences:

1. Not done.
2. X-rays taken 3–22–65 reveal no evidence of metastatic disease.
3. Liver scan 1–29–69 was normal.
4. Chest x-ray of 5–2–72 unchanged, nothing to indicate metastatic disease.
5. 3–2–65 chest film shows clouding along left thorax and pleural thickening.

	TEST						FINDING							
	NO-TEST	TESTN	TESTLOC	VERB-DONE	P	TESTDATE	NEG	BE-SHOW	INDIC	CHANGE	STATUS	MED-FIND	REGION	
													POS	PT-BODY
1.	not	[x-ray]	[chest]	done		[between 12–18–64 & 3–7–65]								
2.		x-rays	[chest]	taken		3–22–65	no	reveal	evidence			(of) metastatic disease		
3.		scan	liver			1–29–69		was			normal			
4.		x-ray	chest		of	5–2–72	neg-prefix			changed [from 11–2–71]				
		[x-ray]	[chest]			[5–2–72]	no-thing	(to) indicate				metastatic disease		
5.		film	chest			3–2–65		shows				clouding	along	(left) thorax
		film	chest			3–2–65		shows				thickening		pleural

[ ] indicates implicit material filled in automatically by regularization after formatting.
( ) indicates left and right adjuncts of main entry.

FIG. 4. Sample formatted sentences from X-ray reports (format has been abridged for presentation).

The word class information—both grammatical and sublanguage—is stored in a lexicon. To obtain the syntactic information, the program first parses the sentence; the parse identifies the necessary syntactic relations: subject-verb-object, host-modifier, etc. Second, it applies standard paraphrastic English transformations to the parsed sentence to regularize the syntactic structure. The most important of these transformations are conjunction expansion, which expands a conjoined noun phrase into two full assertions (see sentence 5 in Fig. 4); and the relative clause transformation, which replaces the relative pronoun by its antecedent: *X-ray shows fracture of right clavicle which might well be pathological fracture → X-ray shows fracture of right clavicle such that fracture might well be pathological fracture.*

The parsing is done by the Linguistic String Project's parser (Grishman et al [12]) and broad coverage English grammar (Sager [13]). The grammar consists of a set of context-free rules augmented by *restrictions* which verify grammatical constraints on tree structure and word class membership. These restrictions are written in Restriction Language (Sager and Grishman [14]), a programming language developed by the Project to express grammatical relations in computer grammars of natural language. For the current application, the grammar was extended to include sentential fragments (Anderson et al. [15]). This extension did not require any major changes to the basic grammar; it primarily involved accepting as sentences certain sequences of constituents already present in the grammar (e.g., noun string, prepositional phrase). The transformations are performed by the Linguistic String Project English transformational component (Hobbs and Grishman [16]). The transformations are written in an extension of Restriction Language and are applied to the output of the parser. The parsing and transformational components were developed to handle general scientific and technical English writing, rather than one particular sublanguage. In consequence, they could be used, with minimal modification, in applications involving different sublanguages.

The third step involves the actual transfer of the parsed regularized sentence into the appropriate format columns. This step is clearly specific to the sublanguage and is implemented as a set of "formatting transformations." These transformations move a word (along with its modifiers) into the appropriate format column. Where there is a one-to-one correspondence between word class and format column, the transformation is very straightforward. For example, since NTEST (e.g., *film*, *X-ray*) occurs only in one location in the format, the TEST-transformation moves any instance of a word of the NTEST class into the TESTN column. Another type of rule involves somewhat more syntactic information: if we find a prepositional phrase whose head noun is for example NBODYPT (e.g., *along the left thorax*) then the preposition *along* will go into the slot POS; on the other hand if the preposition occurs with a date (e.g., *on 10–22–65*), the preposition will go into P under TEST. Thus in order to place a preposition in its correct format slot, we must check the word class of the prepositional object.

The rule for determining negative placement requires still more syntactic information. We must determine whether the negative applies to TEST (and therefore belongs in the NO-TEST column) or to FINDING (in the NEG column):

(1)	*No lesions*	*no* in NEG	(modifies NCONDITION)
	No x-rays	*no* in NO-TEST	(modifies NTEST)
(2)	*No x-rays taken*	*no* in NO-TEST	(occurs with VDONE)
	No x-rays showed abnormality	*no* in NEG	(occurs with VSHOW)

This rule depends not only on what word the negative directly modifies (example 1) but on the word class of the predicate that it occurs with (example 2). The parsed regularized sentence and the word dictionary furnish the syntactic and word class information that the NEG transformation uses to determine the placement of the negative in the format. Five formatted sentences are shown in Fig. 4 (material in square brackets is filled in by the next step).

Once the sentences have been mapped into the format, a final program is applied to check for completeness and consistency of information, and to fill in missing information that is available from context. This final step facilitates the formulation of straightforward retrieval operations because it has the effect of regularizing the information contained in the format. For example, each test performed should have a FINDING; if not, an error message is generated. Similarly each test performed should contain information on the type of test (TESTN), date of test (TESTDATE), and location of test (TESTLOC).

The processing for the TEST columns checks that all of TESTN, TESTLOC, and TESTDATE are filled. If one is missing, it attempts to fill it in from context as follows:

(a) look for the information included in another column of the same entry (e.g., if TESTN = *urinalysis* and TESTLOC is empty, then fill in *urine* for TESTLOC).

(b) otherwise assume continuity of topic within a given report and search backwards through the preceding format lines of the report for the appropriate information to fill in (e.g., in sentence 4 in Fig. 4: *Chest x-ray of 5–2–72 unchanged, nothing to indicate metastatic disease*, fill in the TEST information for the second part (*nothing to indicate metastatic disease*) from TEST information in the first part: TESTN = *x-ray*, TESTLOC = *chest*, TESTDATE = *5–2–72.*)

(c) otherwise fill in a "default" value characteristic of the sublanguage; e.g., if a report consists solely of the sentence *none made* then, since our corpus consists of radiology reports for patients with breast cancer, fill in the default values TESTN = *x-ray*, TESTLOC = *chest*, TESTDATE = *between* date of last report + 1 *and* date of present report (obtained from the report headings).

(d) If one of TEST, TESTLOC, or TESTDATE is still empty, print an error

message (e.g., given the sentence *scan was normal*, there is no default value for TESTLOC when TESTN = *scan*, so the message *TESTLOC missing* is printed).

The referent for referential expressions is also checked and filled in. For example, given the phrase *no change since last chest x-ray*, the program looks for the x-ray referred to by searching backwards through the format entries of preceding *reports* for information referring to a chest x-ray: TESTN = *x-ray* or synonym, TESTLOC = *chest*, TESTDATE less than date of current report, NO-TEST empty (unnegated chest x-ray). If such an entry is found, its date is filled in: *no change since last chest x-ray of 10–22–65*; if no such information is found, an error message is generated. (This process of filling in is done on the formatted sentences; the text is used in the examples for illustrative purposes only.) The material filled in by this process appears in square brackets in the format entries of Fig. 4. The data base is now ready to use.

Fact Retrieval from the Data Base

Once the sentences have been mapped into the format, a program retrieves information from the data base by checking the columns of the format. The entries in a column have the same directionality, e.g., all entries in column NO-TEST indicate negation, and all entries in column CHANGE indicate existence of change: a word like *same* is factored into an entry in NEG and an entry in CHANGE. As a result it is often sufficient just to test whether a column has an entry. For example to answer the question *Was a test done?* the program checks if NO-TEST has an entry; if not, the answer is *YES* (TEST WAS DONE); if NO-TEST has an entry, then it negates the existence of a test and the answer is *NO* (TEST NOT DONE). For other questions the program has to examine the *contents* of a column, e.g., if we wish to know what kind of TEST was done, the answer is the entry in the TESTN column. For more complicated questions, the program must check the co-occurrence of information in several columns: to answer the question *Were any abnormal or suspicious findings reported?* the program makes the following tests:

(1) if NO-TEST has an entry, answer is NOT APPLICABLE (no test done);
(2) else if STATUS has an entry (e.g., *normal*),
  (a) then if NEG is empty (unnegated normal finding), answer is NO,
  (b) else (NEG has an entry, e.g., *not normal*) answer is YES;
(3) else if CHANGE has an entry
  (a) then if NEG is filled (e.g., *no change since report of 1–18–65*) then locate report used as the point of comparison (e.g., *report of 1–18–65*) and use this procedure on it to obtain an answer (since there has been no change)
  (b) else (NEG is empty) answer is YES (e.g., *enlargement of heart*) unless entry in CHANGE indicates improvement;

(4) else if MED-FIND has an entry and NEG is empty (e.g., *lesion in lung*) then answer is YES

(5) else answer is NO.

Using this procedure on the formatted sentences in Fig. 4, the program obtains the following answers for this question:

Question: Were any abnormal or suspicious findings reported?

Sentence	Answer	Test in procedure
1.	NOT APPLICABLE	test 1
2.	NO	test 5
3.	NO	test 2a
4a.	answer from 11–2–71 (not shown here)	test 3a
4b.	NO	test 5
5a.	YES	test 4
5b.	YES	test 4

Question Answering

As an alternative to the hand-coding of retrieval procedures, we have developed a question-answering system which can translate English questions into data base retrieval requests. We expect that these two approaches will be complementary: question-answering will facilitate interactive retrieval of individual facts or simple statistics, while hand-coded procedures will be appropriate for complex tasks.

The job of developing a question-answering system has been considerably simplified by the commonality of linguistic processing between the formatting and question-answering procedures. The vocabulary of the questions and medical records is the same, so a common dictionary can be used. The first two steps in question processing, as in formatting, are parsing and transformational decomposition. The same Linguistic String Project English parser is used, and the grammar is generally the same, although somewhat different English transformations are involved in question analysis than in the formatting.

Once a question has been parsed and transformationally decomposed, it goes through five stages of processing: filtering, anaphora resolution, construction of retrieval request, simplification of retrieval request, and retrieval itself. Although some efficiency might have been gained from merging some of these stages, we believe that clarity is enhanced by keeping them separate. We shall now describe each of these stages briefly in turn.

The *filtering* stage imposes sublanguage selectional constraints; e.g., it checks that a noun is of the proper class to appear as the subject of a particular verb. The filtering resolves syntactic ambiguities which could not be resolved by the general English grammar. This stage also uses the selectional constraints to determine the possible word class membership of any pronoun in the question (for example, that *it* in *When was it taken?* must refer to a *test* in this sublanguage).

The *anaphora resolution* stage seeks to find an antecedent for each pronoun in the question. The present procedure is rather elementary, and is limited to references by pronouns to noun phrases explicitly appearing in prior questions. Prior questions are searched, most recent question first, each question breadth-first top-down left-to-right through the transformational decomposition tree, until a noun phrase is found which would be acceptable (i.e., of the proper word class) in the context of the pronoun. The first such phrase to be found is taken as the antecedent. If the pronoun is indefinite (*one, ones*) it is replaced in the question by its antecedent; if the pronoun is definite (*he, she, it, they*), it is replaced by the name of the referent of the antecedent. For example, in the sequence

Did John have an X-ray?
Did Jane have one?

one will be replaced by *an X-ray*. On the other hand, in the sequence

Did John have an X-ray?
When was it taken?

it will be replaced by the internal identifier assigned to the X-ray retrieved in response to the first question.

After anaphoric references have been resolved, the question is *translated into a retrieval request.* The request is formulated in a first-order predicate calculus, with the predicates expressing conditions on entries in the formatted data base and the quantified variables ranging over the sets of format entries, patients, tests, etc. A calculus of essentially this form, the *relational calculus*, has been suggested by Codd [17] as a suitable query language for relational data base systems. Query languages of similar style have been used as internal representations in other question-answering systems, such as LSNLIS [3].

At the heart of this stage is a procedure for analyzing the quantifier structure of the question. This procedure examines the subject and object(s) of a verb and for each one determines whether it calls for a quantifier. This determination is based primarily on the determiners present in the subject and objects. For example, for the sentence

Did every patient have a chest X-ray?

the procedure will generate a universal quantifier for "*every* patient" and an existential quantifier for "*a* chest X-ray." The procedure incorporates rules for quantifier ordering, so that, in this instance, the quantifier over patients is placed before the quantifier over X-rays. These ordering rules are a consequence of the relationships in this domain: in this case, that one patient may be associated with several X-rays but *not* one X-ray with several patients. (These ordering rules take precedence over surface ordering; for example, *Was a chest X-ray taken of every patient?* should receive the same interpretation even though the surface order has been reversed.)

This procedure is preceded and followed by a series of sublanguage (domain-specific) transformations. These transformations account for paraphrase relations

in the sublanguage, whereas the English transformations which applied earlier account for paraphrase relations in the language as a whole. The English and sublanguage transformations together map the various constructions which can be used in a question into a small set of constructions, each of which corresponds to some subset of columns in the formatted data base. For instance, the following questions are equivalent in meaning (PX is the name of a patient):

Has PX had a chest X-ray?
Has PX had a chest X-ray taken?
Was an X-ray taken of PX's chest?
Was PX's chest X-rayed?

They are all mapped into the structure

take (—, —, an X-ray, PX, chest)

(the two unfilled argument positions would be used to specify the agent performing the test and the test view—lateral or posterior–anterior). This structure is subjected to quantifier analysis, which builds an existential quantifier corresponding to *an* ranging over all tests of type X-ray (or synonyms thereof). After quantifier analysis, the verb *take* with its arguments is converted to an existential quantifier ranging over format rows, with appropriate conditions on the TEST columns:

$$(\exists t \in \text{testidents})(\exists f \in \text{formatset})$$
$$(f_{\text{TESTIDENT}} = t) \wedge (f_{\text{TESTN}} = \text{X-ray}) \wedge (f_{\text{TESTLOC}} = \text{chest}) \wedge (f_{\text{PATIENT}} = \text{PX}).$$

Here f is a variable which takes on as values the rows of the formatted data base. The notation f_X signifies the entry in format column X of the row assigned to f. TESTIDENT is a column added to the data base for use by the question-answering procedures; it associates a unique identifier with each test in the data base.

As the final operation of this stage, the system adds to the retrieval request a *performative function* which will eventually print the answer. For the question *Has PX had a chest X-ray?* an answer of *yes* or *no* would be adequate, but a more complete reply would include a list of PX's chest X-rays, if any. To produce this, the system converts the outermost existential quantifier to a set former and makes the entire expression the argument of TELLIFANY:

$$\text{TELLIFANY}\ (\{t \in \text{testidents} \mid (\exists f \in \text{formatset})$$
$$(f_{\text{TESTIDENT}} = t) \wedge (f_{\text{TESTN}} = \text{X-ray}) \wedge (f_{\text{TESTLOC}} = \text{chest}) \wedge (f_{\text{PATIENT}} = \text{PX})\}).$$

The request is now ready to submit to the retrieval component. However, the multiplicity of quantifiers produced by the translation of a complex question greatly slows down retrieval. To reduce this, the request is first passed through a *simplification stage* which tries to reduce the number of quantifiers without changing the meaning of the request. To simplify the request given above, the system makes

use of the fact that the set *testidents* is the projection of the TESTIDENT column of the format; in other words

$$(\forall f \in \text{formatset})\, f_{\text{TESTIDENT}} \in \text{testidents}.$$

Using this fact, the request can be simplified to

$$\text{TELLIFANY}(\{f_{\text{TESTIDENT}}, f \in \text{formatset} \mid (f_{\text{TESTN}} = \text{X-ray}) \wedge (f_{\text{TESTLOC}} = \text{chest}) \wedge (f_{\text{PATIENT}} = \text{PX})\}).$$

The simplified request is now passed to the *retrieval component*. In our current application, the entire data base can be accommodated in main memory. As a result, it has been possible to implement this component as a simple set of recursive procedures.

An example

Fig. 5 shows the internal representations at several stages during the processing of the dialog:

user: Did every patient have a chest X-ray in 1975?
computer: NO (PZ DID NOT)
user: Has PZ had any since 1975?
computer: NO

Immediately below each question appears the analysis tree following transformational decomposition. If there were a syntactic ambiguity which could not be resolved by the parser, there would be several decomposition trees at this point. In these examples, however, all potential ambiguities were resolved during parsing. The analysis is printed as a nested list structure; the general form of a list for an elementary assertion is

$$(\text{operator subject object adjunct}_1\ \text{adjunct}_2 \ldots)$$

where the operator may be a verb, adjective, or preposition, and the adjuncts are modifiers of the operator (e.g., a time expression modifying a verb). The subject and object may be elementary assertions or noun phrases. A noun phrase has the form

$$(\text{noun or pronoun } (\# n)\ \text{adjunct}_1\ \text{adjunct}_2 \ldots)$$

where n is a serial number assigned to each noun phrase for use in subsequent processing. Thus in the first question the main verb is *have*, with subject *patient* (serial number, 1; number, *singular*; determiner, *every*), object *X-ray* (modified by the assertion (*of X-ray chest*)), adjunct *in* 1975, and tense marker adjunct *past*.

The first stage following the transformational decomposition is the filtering component. Since the parser only produced one parse for each of these questions, the filter's only job is to verify that this analysis meets sublanguage constraints.

The next stage performs anaphora resolution. None is required of the first question. In the second question an antecedent must be found for the word *any* (the parser analyzes *any* as *any* NULLN, where NULLN indicates an omitted noun;

the system then treats NULLN as it would a pronoun). In the sublanguage of X-ray reports, the context *patient has x* implies that x must belong to the NTEST (test noun) or NCONDITION (medical condition noun) class. The anaphora resolution stage searches prior sentences for the first phrase from one of these classes. It finds such a phrase, *chest X-ray*, in the first question, and substitutes it into the tree for the second question. The resulting tree is shown as the second list structure for the second question.

After anaphora resolution comes the generation of the retrieval request. In the first question, *every patient* leads to a universal quantifier over the set of patients, and *a chest X-ray* leads to an existential quantifier over the set of tests, producing the request

$$(\forall x_1 \in \text{patients})\ (\exists x_2 \in \text{testidents})\ (\exists f \in \text{formatset})$$
$$f_{\text{TESTIDENT}} = x_2 \wedge f_{\text{TESTN}} = \text{X-ray} \wedge f_{\text{TESTLOC}} = \text{chest}$$
$$\wedge\ f_{\text{PATIENT}} = x_1 \wedge 1974 < f_{\text{TESTDATE}} < 1976.$$

However, when the performative TELLIFALL is added, the universal quantifier is changed into a set former with a negated condition:

$$\text{TELLIFALL}(\{x_1 \in \text{patients} \mid \neg(\exists x_2 \in \text{testidents})(\exists f \in \text{formatset})$$
$$f_{\text{TESTIDENT}} = x_2 \wedge f_{\text{TESTN}} = \text{X-ray} \wedge f_{\text{TESTLOC}} = \text{chest}$$
$$\wedge\ f_{\text{PATIENT}} = x_1 \wedge 1974 < f_{\text{TESTDATE}} < 1976\}).$$

In other words, the exception set, the set of patients who did not have any X-rays in 1975, is computed. This set will be useful later on in formulating a response.

In the second question, *any chest X-rays* (the phrase reconstructed by anaphora resolution) leads to an existential quantifier over the set of tests. When the performative TELLIFANY is added, the existential quantifier is changed to a set former (no negation is introduced in this case). The requests for both questions are shown in Fig. 5.

For each question, the simplification component is able to eliminate the quantification over the set of tests, just as it did in the example of simplification presented earlier. The simplified requests are also shown in Fig. 5.

Finally, the system generates an answer. In responding to the first question it not only says no, not every patient had an X-ray in 1975, but also—using the exception set it computed—can tell the user specifically that patient PZ did not, prompting the use to ask question 2.

Implementation

The entire system we have described above is running on a Control Data 6600, requiring about 75,000 words of memory. The parsing and transformational stages of the formatting and question-answering are implemented in Restriction Language, a language specially developed for writing natural language grammars (Sager and Grishman [14], Hobbs and Grishman [16]). The remaining stage of the formatting and question-answering are implemented in LISP 1.5.

QUESTION: Did every patient have a chest x-ray in 1975?

TRANSFORMATIONAL DECOMPOSITION:

```
(askwhether
    (have
        (patient (# 1)
                 (T-det (every))
                 (T-number (singular)))
        (X-ray   (# 2)
                 (T-npos
                   (of (X-ray (# 2) (T-number (singular)))
                       (chest (# 3) (T-number (singular)))))
                 (T-det (a))
                 (T-number (singular)))
        (T-pntime (in (have) (1975 (# 4))))
        (T-tense (past))))
```

RETRIEVAL REQUEST:

$$\text{TELLIFALL } (\{x_1 \in \text{patients} \mid \neg(\exists x_2 \in \text{testidents})\ (\exists f \in \text{formatset})\ (f_{\text{TESTIDENT}}=x_2) \wedge (f_{\text{TESTN}}=\text{X-ray}) \wedge (f_{\text{TESTLOC}}=\text{chest}) \wedge (f_{\text{PATIENT}}=x_1) \wedge (1974 < f_{\text{TESTDATE}} < 1976)\})$$

SIMPLIFIED RETRIEVAL REQUEST:

$$\text{TELLIFALL } (\{x_1 \in \text{patients} \mid \neg(\exists f \in \text{formatset})\ (f_{\text{TESTN}}=\text{X-ray}) \wedge (f_{\text{TESTLOC}}=\text{chest}) \wedge (f_{\text{PATIENT}}=x_1) \wedge (1974 < f_{\text{TESTDATE}} < 1976)\})$$

ANSWER: NO (PZ DID NOT)

QUESTION: Has PZ had any since 1975?

TRANSFORMATIONAL DECOMPOSITION:

```
(askwhether
    (have
        (PZ (# 5)
            (T-number (singular)))
        (nulln (# 6)
            (T-det (any)))
        (T-pntime (since (have) (1975 (# 7))))
        (T-aspect (perfect))
        (T-tense (present))))
```

WITH ANAPHORA RESOLVED:

```
(askwhether
    (have
        (PZ (# 5)
            (T-number (singular)))
        (X-ray (# 2)
            (T-npos
                (of (X-ray (# 2) (T-number (singular)))
                    (chest (# 3) (T-number (singular)))))
            (T-det (any))
            (T-number (singular)))
        (T-pntime (since (have) (1975 (# 7))))
        (T-aspect (perfect))
        (T-tense (present))))
```

RETRIEVAL REQUEST:

TELLIFANY ({$x_3 \in$ testidents|($\exists f \in$ formatset)
($f_{\text{TESTIDENT}}=x_3$) $\wedge$ (f_{TESTN}=X-ray) $\wedge$ (f_{TESTLOC}=chest)
$\wedge$ (f_{PATIENT}=PZ) $\wedge$ ($f_{\text{TESTDATE}} > 1975$)})

SIMPLIFIED RETRIEVAL REQUEST:

TELLIFANY ({$f_{\text{TESTIDENT}}$, $f \in$ formatset |
(f_{TESTN}=X-ray) $\wedge$ (f_{TESTLOC}=chest)
$\wedge$ (f_{PATIENT}=PZ) $\wedge$ ($f_{\text{TESTDATE}} > 1975$)})

ANSWER: NO

FIG. 5. Intermediate data structures produced in answering two questions.

Results

The collection of radiology reports used in our initial experiment contained a total of 248 sentences. Of these, 236 (95%) were successfully formatted by the procedures described above. An average of 14.5 seconds of computer time was required to process each sentence. Of this time, 11.5 seconds were consumed in parsing, 2.5 seconds in English and formatting transformations, and 0.5 seconds in normalization. Since these sentences were processed, parsing speed has been significantly increased by improvements in the program and grammar.

Processing times for questions were quite similar. For a small set of test questions, an average of 15 seconds of processor time per question were consumed. This divided into 10.4 seconds for parsing, 2.1 seconds for English transformations, and 2.5 seconds for the remaining stages, including retrieval.

Future plans

It has been our goal to develop techniques for automatic data base creation that would apply to any sublanguage; that is, to any set of documents drawn from a specific subject matter area which deals with a restricted number of topics and uses a corresponding subset of English vocabulary and syntax.

We are currently applying these techniques to a much more complex sublanguage, namely hospital discharge summaries. We have established the sublanguage word classes for this material and have developed a new format by manual analysis. The format contains several times as many columns as the format for the radiology material, reflecting the wider range of topics discussed in the discharge summaries: patient state, quantitative tests and results, treatments, doctor's actions. The format also allows for a detailed treatment of time and aspectual expressions. The syntax of the discharge summaries is more complex, requiring a larger set of English transformations for regularization. The formatting component is also larger due to the increase in the size of the format.

The most significant difference for the discharge summaries is the importance of relative chronology within a single report. Extensive regularization after formatting is required to assign a partial ordering over time to the individual entries. The

ordering is needed to answer retrieval requests such as *Was a blood culture taken at admission?* or *Did the symptoms persist?* From our work on discharge summaries it is clear that the more complex the interconnections between items in the document, the more complex the format becomes (in order to represent these relations—such as time), and the more complex the regularization after formatting. This stage (regularization after formatting) processes all information about intersentential relations; the preceding stages operate only on isolated sentences. Preliminary results indicate that over 50% of the total processing time is spent in this stage of regularization for the discharge summaries, as compared to less than 5% for the radiology material.

The question-answering component is still quite rudimentary. It does not yet cover all the columns of the radiology data base, and, for the data base relations it does cover, accepts only a limited set of question formulations. Thus, to make the system practical, substantial expansion is still required. Most of this expansion will come in the stage which generates the retrieval requests; additional transformations are needed here to map the various constructions which a questioner might use into the relations of the data base.

ACKNOWLEDGMENTS

This investigation was supported in part by NIH Grant LM 02616 from the National Library of Medicine, in part by Research Grant No. SIS–75–22945 from the National Science Foundation, Division of Science Information, in part by contract N00014–75–C–0571 with the Office of Naval Research, and in part by ERDA Administration Contract EY–76–C–02–3077*000. The final stages of both the formatting and the question-answering were programmed by Jim Litsas.

REFERENCES

1. Simmons, R. F., Natural language question-answering systems: 1969, *Comm. ACM* **13** (January, 1970) 15–30.
2. Petrick, S. R., On natural language based computer systems, *IBM J. Res. Devel.* **20** (July, 1976) 314–325.
3. Woods, W. A., Kaplan, R. M. and Nash-Webber, B., The lunar sciences natural language information system; final report, *BBN Report 2378*, Bolt, Beranek and Newman, Inc., Cambridge, MA (1972).
4. Plath, W. J., REQUEST: A natural language question-answering system, *IBM J. Res. Devel.* **20** (July, 1976) 326–335.
5. Waltz, D. J. (Ed.), Natural Language Interfaces, *ACM SIGART Newsletter* **61** (February, 1977) 16–65.
6. Morgan, J. D., Computerized SNOMED encoding: a practical solution, in: Shires, D. B. and Wolf, H., (Eds.), *MEDINFO 77*, (North-Holland Publ. Co., Amsterdam, 1977) 277–281.
7. Dunham, G. S., Pacak, M. G. and Pratt, A. W., Automatic indexing of pathology data, *Journal of the American Society for Information Science* **29** (2) (March 1978) 81–90.
8. Sager, N., Syntactic formatting of scientific information, *Proceedings of the 1972 Fall Joint Computer Conference*, AFIPS Conference Proceedings, Vol. **41** (AFIPS Press, Montvale, NJ, 1972) 791–800.

9. Hirschman, L., Grishman, R. and Sager, N., From text to structured information: automatic processing of medical reports, *Proceedings of the 1976 National Computer Conference*, AFIPS Conference Proceedings, Vol. **45** (AFIPS Press, Montvale, NJ, 1976) 267–275.
10. Sager, N. and Lyman, M., Computerized language processing: implications for health care evaluation, *Medical Record News, Journal of the American Medical Record Association* **49** (3) (June 1978) 20–30.
11. Hirschman, L., Grishman, R. and Sager, N., Grammatically-based automatic word class formation, *Information Processing and Management* **11** (1975) 39–57.
12. Grishman, R., Sager, N., Raze, C. and Bookchin, B., The linguistic string parser, *Proceedings of the 1973 Computer Conference*, AFIPS Conference Proceedings, Vol. **42** (AFIPS Press, Montvale, NJ, 1973) 427–434.
13. Sager, N., Natural language information processing: a computer grammar of English and its applications, to appear.
14. Sager, N. and Grishman, R., The restriction language for computer grammars of natural language, *Comm. ACM* **18** (1975) 390–400.
15. Anderson, B., Bross, I. D. J. and Sager, N., Grammatical compression in notes and records: analysis and computation, paper delivered at the 13th Annual Meeting of the Association of Computational Linguistics, Boston (November 1 1975) *American Journal of Computational Linguistics*, **2** (4) (1975).
16. Hobbs, J. and Grishman, R., The automatic transformational analysis of English sentences: an implementation, *International Journal of Computer Mathematics, A*, **5** (1976) 267–283.
17. Codd, E. F., Relational completeness of data base sublanguages, in: Rustin, R. (Ed.), *Courant Computer Science Symposium 6: Data Base Systems*, (Prentice–Hall, Englewood Cliffs, NJ, 1972) 65–98.