MULTIPLE REPRESENTATIONS OF KNOWLEDGE FOR TUTORIAL REASONING

John Seely Brown and **Richard R. Burton**
Bolt Beranek and Newman
Cambridge, Massachusetts

I. INTRODUCTION

This chapter provides an overview of SOPHIE, an "intelligent" instructional system which reflects a major attempt to extend Carbonell's notion of mixed-initiative Computer Aided Instruction [introduced in SCHOLAR (Carbonell, 1970)] for the purpose of encouraging a wider range of student initiatives. Unlike previous AI-CAI systems which attempt to mimic the roles of a human teacher, SOPHIE tries to create a "reactive" environment in which the student learns by trying out his ideas rather than by instruction. To this end, SOPHIE incorporates a "strong" model of its knowledge domain along with numerous heuristic strategies for answering a student's questions, providing him with critiques of his current solution paths and generating alternative theories to his current hypotheses. In essence, SOPHIE enables a student to have a one-to-one relationship with an "expert" who helps the student create, experiment with, and debug his own ideas.

SOPHIEs expertise is derived from an efficient and powerful inferencing scheme that uses multiple representations of knowledge including (i) simulation models of its microcosm, (ii) procedural specialists which contain logical skills and heuristic strategies for using these models, and (iii) semantic nets for encoding time-invariant factual knowledge. The power and generality of SOPHIE stems, in part, from the synergism obtained by focusing the diverse capabilities of the procedural specialists on the "intelligent" manipulation, execution, and interpretation of its simulation models. In this respect SOPHIE represents a departure from current inferencing paradigms (of either a procedural or declarative nature) which use a uniform representation of information.

Before delving into any details about SOPHIE, we first present the basic scenario which shaped SOPHIEs outward appearance and which defined the kinds of logical and linguistic tasks it had to be able to perform. We then provide an annotated example of a student using SOPHIE followed by a discussion of its natural language processor. SOPHIEs language processor is still in its infancy; its primary interest lies in the use of a semantic "grammar" to successfully cope with the nasty problems of anaphoric references, deletions and complex ellipses inherent in any realistic man-machine dialog. We then describe the specialized inferencing techniques and the multiple representations of knowledge embodied in SOPHIE.

A. Basic Scenario

In the basic scenario, SOPHIE acts as an electronics lab instructor who helps the student transform his classroom knowledge of electronics into an experiential, intuitive knowledge of its meaning and application. It does this by interacting with the student while he is debugging a malfunctioning piece of equipment.[1] The student can perform any sequence of measurements, ask either specific questions about the implications of these measurements or more general hypothetical questions, and even ask for advice about what to consider next, given what he has discovered thus far. At any time SOPHIE may encourage the student to make a guess as to what he thinks might be wrong given the measurements he has made thus far. If he does, SOPHIE will evaluate his hypothesis by taking into consideration all the information he should have been able to derive from his current set of measurements. If any of this information is logically contradicted by the hypothesis, SOPHIE identifies and explains these contradictions. Likewise SOPHIE can judge the merits of any particular measurement with respect to the prior sequence of measurements he has made. For example, his new measurement may be logically redundant in the sense that no new information can possibly be derived from it (an extremely complex task to determine). SOPHIE can also decide if this measurement performs a reasonable split of the hypotheses space of possible faults which have not yet been ruled out by prior measurements.

It should be noted that the preceeding scenario contains quite a variety of logical tasks (i.e., hypothesis evaluation, hypothesis formation, redundancy checking, hypothetical question answering) each one of which requires a substantial amount of deep logical inferencing. One of the basic challenges in constructing SOPHIE was creating an inference system which could perform this wide range of tasks efficiently (so that it could be used in real time) and at the same time have it be robust in the sense of handling all realistic queries.

---

[1] Although throughout this chapter the domain of knowledge under consideration is electronics, the reasoning and linguistic paradigms underlying SOPHIE are applicable to many domains outside of electronics.

B. Protocol

The following protocol reveals some of the linguistic and logical capabilities of our current version of SOPHIE. We have included numerous annotations in the protocol thereby (hopefully) making it understandable to those readers not grounded in electronic jargon. In fact, simply from a linguistic point of view it is important to point out where many subtle problems arise due to anaphora and ellipsis, where domain specific "default" assignments must be made and yet might go unnoticed by a reader unfamiliar with electronics.

```
WELCOME TO SOPHIE, A REACTIVE LEARNING
ENVIRONMENT. THE PIECE OF EQUIPMENT YOU
ARE STUDYING IS A REGULATED POWER
SUPPLY.[2] WOULD YOU LIKE TO START WITH A
FAULTED INSTRUMENT WHICH NEEDS FIXING?
>>YES[3]
```

SOPHIE randomly selects a fault, inserts it into a model of the instrument and tells the student how the controls are set.

```
THE INSTRUMENT HAS A FAULT AND ITS PRESENT
CONTROL SETTINGS ARE:

CC      1.0     CURRENT CONTROL
CR      HIGH    CURRENT RANGE SWITCH
LOAD    20      LOAD RESISTANCE IN OHMS
VC      1.0     VOLTAGE CONTROL
VR      HIGH    VOLTAGE RANGE SWITCH
```

It is assumed that the student has already had an introduction to the basic principles of the power supply so

---

[2] The schematic is shown in Fig. 1.

[3] The student's typing is shown in bold italics.

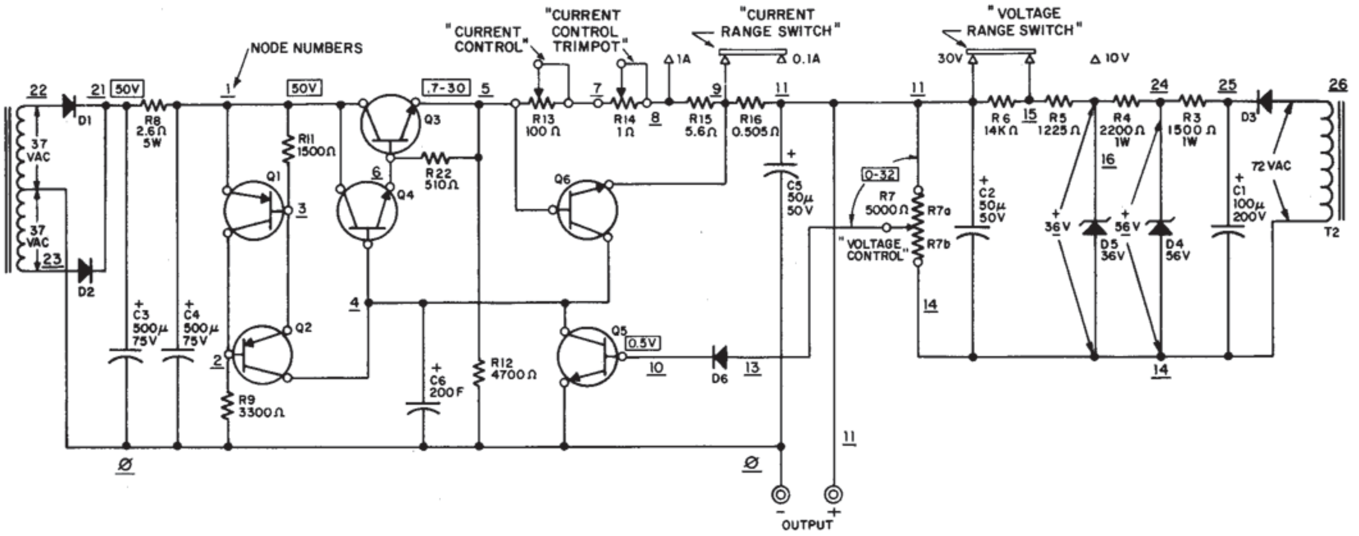

Fig. 1. Schematic diagram of circuit. The circuit is based the Heathkit IP-28 power supply which is a reasonably sophisticated power supply with both current limiting and voltage limiting behavior. These two interrelated feedback loops greatly complicate the troubleshooting process.

the above comments are sufficient to characterize the controls.

>>*WHAT IS THE OUTPUT?*
THE OUTPUT VOLTAGE IS 11.7 VOLTS.

The student begins his troubleshooting by performing measurements in the faulted circuit. Notice that "the output" was disambiguated as the output voltage.

>>*WHAT IS IT IN A WORKING INSTRUMENT?*
IN A WORKING CIRCUIT THE OUTPUT VOLTAGE IS 19.9 VOLTS.

In case the student does not know what the value of a measurement should have been, he can have any measurement repeated in a working instrument. Note the student's use of the pronoun "it" to refer to the previous measurement.

>>*WHAT IS THE VOLTAGE BETWEEN R8 AND Q1?*
THE VOLTAGE AT N1 IS 47.6 VOLTS.

The student requests an internal measurement. Notice that even though the sentence form is "voltage between x and y", it really means "voltage at z with respect to ground." The output routines reduce the input to its simpler form (i.e., between x and y reduces to Node 1).

>>*WHAT IS THE BSE EMITTER VOLTAGE OF THE VOLTAGE LIMITING TRANSITOR?*
=BASE
=TRANSISTOR
THE BASE EMITTER VOLTAGE OF Q5 IS -.037 MILLIVOLTS.

The spelling corrector has detected and corrected two misspellings. The system must also know that Q5 is the voltage limiting transistor.

>>*WHAT ABOUT FOR THE CURRENT LIMITING TRANSISTOR?*

THE BASE EMITTER VOLTAGE OF Q6 IS .349 VOLTS.

The meaning of this question is completely dependent on the context established by the prior sentence. The system uses this context to determine that he is interested in not only voltage measurements but the voltage across a particular junction.

>>*WHAT SHOULD IT BE?*
IN A WORKING CIRCUIT THE BASE EMITTER VOLTAGE OF Q6 IS .593 VOLTS.

Another way of requesting that the prior measurement be repeated in a working instrument, i.e., the meaning of "should" implies that a working instrument be used.

>>*REPLACE Q6*
I AM GOING TO ASK YOU SOME QUESTIONS ABOUT HOW Q6 IS FAULTED.

The student believes he has discovered the problem and tries to fix it. Before a component is replaced, however, the system queries him as to how he thinks it is faulted. In the case of transistors (which have complex fault modes), if he gets at least something right, the system tells him what was actually wrong and replaces the broken transistor.

ARE ANY JUNCTIONS SHORTED?
>>*I DON'T KNOW*
ARE ANY TERMINALS OPEN?
>>*YES*
WHICH TERMINAL(S) ARE OPEN?
>>*THE BASE*
Q6 IS NOT FAULTED IN THAT MANNER. Q6 NOT REPLACED.

After learning that Q6 does not have an open base, the student asks if this was at least reasonable hypothesis.

>>*IS IT POSSIBLE THAT Q6 HAS AN OPEN BASE?*

Before the system's evaluation procedures can be applied to his idea or hypothesis, the hypothesis must be completely specified. To help the student out, the system suggests a plausible complete specification of it.

```
IS THE COMPLETE FAULT MODE EB/EC/BC =
OP/OK/OP WHAT YOU INTENDED?
>>SURE
```

The fault mode should be read as: the emitter base junction (EB) is open (OP), the emitter collector junction (EC) is okay (OK) and the base collector junction (BC) is open (OP).

```
WHEN THE SETTINGS WERE:
CC=1.0, CR=HIGH, LOAD=20, VC=1.0, VR=HIGH
THAT FAULT CONFLICTS WITH THE FOLLOWING
MEASUREMENT.
YOU OBSERVED THAT THE OUTPUT VOLTAGE
WAS 11.7 VOLTS.
IF Q6 HAD FAULT MODE EB/EC/BC = OP/OK/OP
IT WOULD BE 23.9 VOLTS.
IN A WORKING CIRCUIT IT IS 19.9 VOLTS.
```

In determining the logical consistency of his hypothesis with respect to the measurements he has already made, a glaring counterexample is discovered.

```
ALSO:
THAT FAULT CONFLICTS WITH THE FOLLOWING
MEASUREMENT.
YOU OBSERVED THAT THE BASE EMITTER
VOLTAGE OF Q6 WAS .349 VOLTS.
IF Q6 HAD FAULT MODE EB/EC/BC = OP/OK/OP
IT WOULD BE .712 VOLTS.
IN A WORKING CIRCUIT IT IS .593 VOLTS.
```

The system then points out a more subtle counterexample.

```
THAT FAULT DOES NOT EXPLAIN ANY OF THE
SYMPTOMS YOU OBSERVED BUT IT IS
CONSISTENT WITH THE VALUES FOR THE BASE
```

> EMITTER VOLTAGE OF Q5 AND THE VOLTAGE AT N1.

Finally the system tells him what is right or at least consistent with his hypothesis.

> >>*WHAT IS THE OUTPUT OF THE CONSTANT CURRENT SOURCE?*
> THE OUTPUT OF THE CONSTANT CURRENT SOURCE IS -.59 MILLIAMPS.

To understand this request the system must also know about the functional blocks of the instrument. Also note that the system had to determine that "output" in this context meant "output current" as opposed to "output voltage" as was meant in the first question of the protocol!

> >>*V BETWEEN N4 AND N5*
> THE VOLTAGE BETWEEN N4 AND N5 IS 1.64 VOLTS.

Noun phrase utterances get interpreted as questions. "V" is an accepted abbreviation for voltage.

> >>*COULD THE PROBLEM BE THAT THE BASE OF Q3 IS OPEN?*

The student constructs another hypothesis

> IS THE COMPLETE FAULT MODE EB/EC/BC = OP/OK/OP WHAT YOU INTENDED?
> >>*YES*
> UNDER THE SETTINGS:
> CC=1.0, CR=HIGH, LOAD=20, VC=1.0, VR=HIGH
> THE FAULT WHICH YOU PROPOSED WILL PROPAGATE.
> LIKELY FAULTS WHICH MIGHT OCCUR ARE:
> R22 OPEN
> WOULD YOU LIKE ME TO CONTINUE TESTING YOUR HYPOTHESIS IN A CIRCUIT IN WHICH R22 IS OPEN ALSO?
> >>*NO*

The hypothesis proposed by the student logically entails another component being blown. Before further evaluation of his hypothesis, the user is queried as to whether he realizes this entailment. If he does not, he is given the chance to have his hypothesis evaluated in a "ghost" world in which that component would not have blown.

```
WOULD YOU LIKE ME TO COMMENT ON YOUR
HYPOTHESIS AS IF NO OTHER PARTS HAD
BLOWN?
>>YES
WHEN THE SETTINGS WERE:
CC=1.0, CR=HIGH, LOAD=20, VC=1.0, VR=HIGH,
THAT FAULT WOULD HAVE CAUSED THE
FOLLOWING SYMPTOM.
YOU OBSERVED THAT THE VOLTAGE BETWEEN
N4 AND N5 WAS 1.64 VOLTS.
IF Q3 HAD FAULT MODE EB/EC/BC = OP/OK/OP
IT WOULD BE 31.3 VOLTS.
IN A WORKING CIRCUIT IT IS 1.7 VOLTS.

ALSO:
THAT FAULT CONFLICTS WITH THE FOLLOWING
MEASUREMENT.
YOU OBSERVED THAT THE OUTPUT VOLTAGE
WAS 11.7 VOLTS.
IF Q3 HAD FAULT MODE EB/EC/BC = OP/OK/OP
IT WOULD BE 1.19 VOLTS.
IN A WORKING CIRCUIT IT IS 19.9 VOLTS.

THAT FAULT DOES NOT EXPLAIN ANY OF THE
SYMPTOMS YOU OBSERVED BUT IT IS
CONSISTENT WITH THE VALUES FOR THE
OUTPUT CURRENT OF THE CONSTANT CURRENT
SOURCE, THE BASE EMITTER VOLTAGE OF Q5
AND THE VOLTAGE AT N1.
```

Again the student's hypothesis fails.

```
>>SAVE
CONTEXT SAVED, TO RETURN TO IT TYPE
'RESTORE'.
```

```
YOU NOW HAVE A WORKING INSTRUMENT
WITH WHICH TO EXPERIMENT.
```

The student has discovered more flaws in his understanding and is probably surprised by the high voltage between N4 and N5 that follows from his hypothesis (i.e., 31.3 volts). Wishing perhaps to explore what caused R22 to blow and what led to this unexpectedly high voltage he requests that the present troubleshooting context be saved (i.e., his fault and all his measurements which are the gist for the evaluation specialist). He can now perform any experiments he wishes.

```
>>WHAT IS THE CURRENT THRU R22 WHEN
IT IS SHORTED?
IF R22 WERE SHORTED UNDER THE PRESENT
SETTINGS, THE CURRENT THRU R22 WOULD BE
61 MILLIAMPS.
IN A WORKING CIRCUIT IT IS 1.8 MILLIAMPS.
```

This is an example of a hypothetical or conditional question. Note that "it" refers to R22 which is embedded in a noun phrase earlier in the same statement.

```
>>WHAT ARE THE SPECS OF R22?
THE SPECIFICATIONS OF R22 ARE:
VALUE -- 510 OHMS
POWER/RATING -- .5 WATTS
```

He now asks a factual question which simply gets looked up in the semantic net.

```
>>RESTORE
PREVIOUS SITUATION RESTORED.
```

He may have figured out why R22 blew when Q3 opened and now wishes to get back to the task of troubleshooting.

```
>>WHAT COULD BE WRONG?
HERE ARE SOME FAULTS WHICH EXPLAIN THE
MEASUREMENTS YOU OBSERVED:
THE BETA OF Q3 IS LOW
THE BETA OF Q4 IS LOW
```

OR R22 HAS THE WRONG RESISTANCE

He seems to have run out of ideas and requests some help. At this point the hypothesis generation "specialists" are invoked. These procedures examine all the measurements he has thus far made and then construct a collection of hypotheses which are logically consistent with the values of these measurements.

>>*IS THE EMITTER CURRENT OF Q4 CORRECT?*
THE EMITTER CURRENT OF Q4 IS -61 MILLIAMPS.
IN A WORKING CIRCUIT THE EMITTER CURRENT OF Q4 IS -49 MILLIAMPS.
THAT VALUE SEEMS TO BE INCORRECT.

This question is in some sense a cheat. It really consists of two measurements. If SOPHIE were assigning costs to his measurements or questions he would be charged accordingly.

>>*COULD R22 BE LOW?*
IS IT OKAY IF I USE 25 OHMS FOR THE RESISTANCE OF R22?
>>*YES*
THAT FAULT AGREES WITH ALL OF THE MEASUREMENTS YOU HAVE MADE.

Finally he generates a reasonable hypothesis, but this does not mean that R22 is necessarily low! Just that R22 being low is consistent with what he should be able to determine from his measurements thus far.

>>*WHAT IS THE RESISTANCE OF R22?*
THE MEASURED RESISTANCE OF R22 IS 25 OHMS.
>>*REPLACE R22*
R22 REPLACED
NOW YOU SHOULD TAKE MEASUREMENTS TO SEE IF THERE IS ANYTHING ELSE WRONG.

The system decides he must know what is wrong with R22 so it does not hassle him like it did when he asked for Q6 to be replaced.

```
>>NOW WHAT IS THE OUTPUT?
THE OUTPUT VOLTAGE IS 19.9 VOLTS.
>>WITH A 1000 OHM LOAD?
WITH THE LOAD NOW SET TO 1000
THE OUTPUT VOLTAGE IS 30 VOLTS.
```

He discovers that the unit is now functioning correctly. Note the ability of the system to handle rather extensive ellipses such as "with a 1000 ohm load".

```
>>WHAT WOULD HAPPEN IF R22 WERE
SHORTED?
IF R22 WERE SHORTED UNDER THE PRESENT
SETTINGS:
CC=1.0, CR=HIGH, LOAD=1000, VC=1.0, VR=HIGH
THERE WOULD BE NO CHANGE IN THE OUTPUT
VOLTAGE (I.E.  30 VOLTS).
HOWEVER UNDER THE SETTINGS:
CC=1.0, CR=HIGH, LOAD=10, VC=1.0, VR=LOW,
THE OUTPUT VOLTAGE WOULD BE .623 VOLTS.
IN A WORKING CIRCUIT IT IS 9.93 VOLTS.
```

A good example of what else is involved in handling hypothetical questions. Here the system first guesses that he is interested in output voltage since he didn't specify anything particular. Then it fails to find any change of the output voltage with the present load resistance. However, before it concludes that R22 has no effect it uses some other knowledge about the circuit to decide that a better test case or example would be to set the instrument up with a heavier load. Under this setting it does find a significant change.

```
>>GOODBYE
```

II. NATURAL LANGUAGE UNDERSTANDING

Since we are trying to build an environment in which students will create and articulate ideas, we must provide a convenient way for them to communicate these ideas to the system. A student will become very frustrated if, after forming an idea, he has to try several ways of expressing it before the system understands it. In addition he will become bored if there is a long delay (say 10 seconds) before he gets a response. To compound our problem we discovered from using early versions of SOPHIE that when a person communicates with a logically "intelligent" system he inevitably starts to assume that the system shares his "world-view" or is at least "intelligent" in the linguistic art of following a dialog. In other words, SOPHIE had to cope with problems such as anaphoric references, context-dependent deletions, and ellipses which occur naturally in dialogs. In fact handling these constructs seemed more important than building a system endowed with great syntactic paraphrase capabilities.

A. Semantic Grammar

These realizations led us to build a natural language processor based on the concept of a semantic grammar. A semantic grammar captures some of the simpler notions of conceptual dependencies, enabling one to predict on semantic grounds the referent of an anaphoric expression or the element which has been ellipsed or deleted. In a semantic grammar the usual syntactic categories such as noun, noun phrase, verb phrase, etc. are replaced by semantically meaningful categories. These semantic categories represent conceptual entities known to the system such as "measurements", "circuit elements", "transistors", "hypotheses", etc. (While such refinement can lead to a phenomenal proliferation of nonterminal categories in a grammar, the actual number is limited by the number of underlying concepts which can be discussed. For SOPHIEs present domain, there are on the order of 50 such concepts.)

The grammar which results from this refinement is a formal specification of constraints between concepts. That is, for each concept there is a grammar rule which explicates the ways of expressing that concept in terms of its constituent concepts. Each rule also provides explicit information concerning which of its constituent concepts can be deleted or pronominalized. Once the dependencies have formalized into the semantic grammar, each rule in the grammar is encoded (by hand) as a LISP procedure. This encoding process imparts to the grammar a top-down control structure and specifies the order of application of the various alternatives of each rule. The resulting collection of LISP functions constitute a goal-oriented parser in a fashion similar to SHRDLU (Winograd, 1972).

Encoding the grammar as LISP procedures shares many of the advantages which ATNs[4] (Woods, 1970) have over using traditional phrase structure grammar representations. Four of these advantages are:

(i) the ability to collapse common parts of a grammar rule while still maintaining the perspicuity of the semantic grammar,

(ii) the ability to collapse similar rules by passing arguments (as with SENDR),

(iii) the ease of interfacing other types of knowledge (in SOPHIE, primarily the semantic network) into the parsing process, and

(iv) the ability to build and save arbitrary structures during the parsing process.

In addition to the advantages it shares with ATN representation, the LISP encoding has the computational advantage of being compilable directly into efficient machine code.

Result of the Parsing: Basing the grammar on conceptual entities eliminates the need for a separate semantic interpretation phase. Since each of the nonterminal categories in the grammar is based on a semantic unit, each

---

[4] All of these advantages are, of course, also shared by a PROGRAMMAR grammar (Winograd, 1972).

rule can specify the semantic description of a phrase that it recognizes in much the same way that a syntactic grammar specifies a syntactic description. Since the rules are encoded procedurally, each rule has the freedom to decide how the semantic descriptions returned by the constituent items of that rule are to be put together to form the correct "meaning".

For example, the meaning of the phrase "Q5" is just Q5. The meaning of the phrase "the collector of Q5" is (COLLECTOR Q5) where COLLECTOR is a function encoding the meaning of "collector". "The voltage at the collector of Q5" becomes (MEASURE VOLTAGE (COLLECTOR Q5)) where MEASURE is the procedural specialist who knows about the concept of a measurement. The relationship between a phrase and its meaning can be straightforward and, if the concepts and the specialists in the query language are well matched, usually is. It can get complicated, however. Consider the phrases "the base emitter of Q5 shorted" and "the base of Q5 shorted to the emitter". The thing which is "shorted" in both of these phrases is the "base emitter junction of Q5." The rule which recognizes both of these phrases, PART/FAULT/SPEC, can handle the first phrase by invoking its constituent concepts of JUNCTION (base emitter of Q5) and FAULT/TYPE (shorted) and combining their results. In the second phrase, however, it must construct the proper junction from the separate occurrences of the two terminals involved. Notice that the parser does some paraphrasing, as the "meaning" of the two phrases is the same.

The result returned by the parser is the "meaning" of the entire statement in terms of a simple program. This program specifies which of the procedural specialists should be called (and in what order) to calculate an answer to the student's question or perform the student's command. It is also used by the output generation routines to construct an appropriate phrasing of the response.

B. Use of Semantic Information During Parsing

Prediction: Having described the notion of a semantic grammar, we now describe the ways it allows semantic information to be used in the understanding process. One use of semantic grammars is to predict the possible alternatives that must be checked at a given point. Consider for example the phrase "the voltage at xxx" (e.g., "the voltage at the junction of the current limiting section and the voltage reference source"). After the word "at" is reached in the top-down, left-right parse, the grammar rule corresponding to the concept "measurement" can predict very specifically the conceptual nature of "xxx", i.e., it must be a phrase specifying a location (node) in the circuit.

This predictive information is also used to aid in the determination of referents for pronouns. If the above phrase were "the voltage at it", the grammar would be able to restrict the class of the possible referents to locations. By taking advantage of the available sentence context to predict the semantic class of possible referents, the referent determination process is greatly simplified. For example:

(1a) Set the voltage control to .8?
(1b) What is the current thru R9?
(1c) What is it with it set to .9?

In (1c), the grammar is able to recognize that the first "it" refers to a measurement (that the student would like re-taken under slightly different conditions). The grammar can also decide that the second "it" refers to either a potentiometer or to the load resistance (i.e., one of those things which can be set.). The referent for the first "it" is the measurement taken in (1b), the current thru R9. The referent for the second "it" is "the voltage control" which is an instance of a potentiometer. The context mechanism which selects the referents will be discussed later.

Simple Deletion: The semantic grammar is also used to recognize simple deletions. The grammar rule for each conceptual entity knows the nature of that entity's constituent concepts. When a rule cannot find a constituent concept, it can either

(i) fail (if the missing concept is considered to be obligatory in the surface structure representation), or

(ii) hypothesize that a deletion has occurred and continue.

For example, the concept of a TERMINAL has (as one of its realizations) the constituent concepts of a TERMINAL-TYPE and a PART. When its grammar rule only finds the phrase "the collector", it uses this information to posit that a part has been deleted (i.e., TERMINAL-TYPE gets instantiated to "the collector" but nothing gets instantiated to PART). SOPHIE then uses the dependencies between the constituent concepts to determine that the deleted PART must be a TRANSISTOR.

Ellipses: Another use of the semantic grammar allows the processor to accept elliptic utterances. These are utterances which do not express complete thoughts (i.e., a completely specified question or command) but only give differences between the underlying thought and an earlier one.[5] For example, (2b) and (2c) are elliptic utterances.

(2a) What is the voltage at Node 5?
(2b) At Node 1?
(2c) What about between nodes 7 and 8?

There is a grammar rule for elliptic phrases which is aware of which constructs are frequently used to contrast similar complete thoughts and recognizes occurrences of these as ellipses. This grammar rule identifies which concept or class of concepts are possible from the context available in a elliptic utterance. Later we will discuss the mechanism that determines to which complete thought an ellipsis refers.

C. Using Context to Determine Referents

Pronouns and Deletions: Once the parser has determined the existence and class (or set of classes) of a pronoun or deleted object, the context mechanism is invoked to determine the proper referent. This mechanism has a history of student interactions during the current session which contains for each interaction the parse (meaning) of the student's statement and the response calculated by the

---

[5] This is not strictly the standard use of the word "ellipsis."

system. This history list provides the range of possible referents and is searched in reverse order to find an object of the proper semantic class (or one of the proper classes). To aid in the search of the history list, the context mechanism knows how each of the procedural specialists which can appear in a parse uses its arguments. For example, the specialist MEASURE has a first argument which must be a quantity and a second argument which must be a part, a junction, a section, a terminal, or a node. Thus when the context mechanism is looking for a referent which can either be a PART or a JUNCTION, it will look at the second argument of a call to MEASURE but not the first. Using the information about the specialists, the context mechanism looks in the present parse and then in the next most recent parse, etc. until an object from one of the specified classes is found.

The significance of using the specialist to filter the search instead of just keeping a list of previously mentioned objects is that it avoids misinterpretations due to object-concept ambiguity. For example, the object Q2 is both a part and a transistor. If the context mechanism is looking for a part, Q2 will be found only in those sentences in which it is used as a part and not in those in which it is used as a transistor. In this way the context mechanism finds the most recent occurrence of an object being used as a member of one of the recognized classes.

Referents for Ellipses: If the problem of pronoun resolution is looked on as finding a previously mentioned object for a currently specified use, the problem of ellipsis can be thought of as finding a previously mentioned use for a currently specified object. For example,

(3a) What is the base current of Q4?
(3b) In Q5?

The given object is "Q5" and the earlier function is "base current". For a given elliptic phrase, the semantic grammar identifies the concept (or class of concepts) involved. In (3b), since Q5 is a transistor, this would be TRANSISTOR. The context mechanism then searches the history list for a specialist in a previous parse which accepts the given class as an argument. When one is found, the new phrase is substituted into the proper argument position and the substituted meaning is used as the meaning of the ellipsis. Currently recognition of ellipsed information proceeds in a top-down fashion. In a domain which has many possible concepts used in ellipses, the recognition of the ellipsed concept should either proceed bottom-up or be restricted to concepts recently mentioned.

D. Fuzziness

Having the grammar centered around semantic categories allows the parser to be sloppy about the actual words it finds in the statement. This sense of having a concept in mind, and being willing to ignore words to find it, is the essence of keyword parsing schemes. It is effective in those cases where the words that have been skipped are either redundant or specify gradations of an idea which are not distinguished by the system. Semantic grammars provide the ability to blend keyword parsing of those concepts which are amenable to it with the structual parsing required by more complex concepts.

The amount of sloppiness (i.e., how many (if any) words in a row can be ignored) is controlled in two ways. First, whenever a grammar rule is invoked, the calling rule has the option of limiting the number of words that can be skipped. Second, each rule can decide which of its constituent pieces or words are required and how tightly controlled the search for them should be. Taken together, these controls have the effect that the normal mode of operation of the parser is tight in the beginning of a sentence but more fuzzy after it has made sense out of something.

E. Results

Our two goals for SOPHIEs natural language processor are efficiency and friendliness. In terms of efficiency, the parser has succeeded admirably. The grammar written in INTERLISP (Teitelman, 1974) can be block compiled.

Using this technique, the complete parser takes up about 5k of storage and parses a typical student statement consisting of 8 to 12 words in around 150 milliseconds!

Our goal of friendliness is much harder to measure since the only truly meaningful evaluation must be made when students begin using SOPHIE in the classroom. Our results so far, however, have been encouraging. The system has been used in hundreds of hours of tests by people involved in the SOPHIE project. In addition, several dozen different people have had realistic sessions (as opposed to demonstrations) with SOPHIE and the parser was able to handle most of the questions which were asked. Anytime a statement is not accepted by the parser, it is saved on a disk file. This information is constantly being used to update and extend the grammar.

F. Expanding the Natural Language Processor

Areas in which the natural language processor is lacking at present include relative clauses, quantifiers, and conjunctions -- the most noticeable being the lack of conjunction. While incorporating conjunction in a systematic way will almost certainly require an additional mechanism, the semantic nature of our nonterminal categories and the predictive ability of the semantic grammar should provide a good handle on the combinatorial explosion normally accompanying conjunction.

Another area in which the semantic grammar looks especially useful is in providing constructive feedback to the user when one of his statements fails to parse. Such feedback is very important for without it the user does not know whether to try rephrasing his question (if so, how) or to give up altogether on this line of questioning. In general, when a statement is not accepted by our top-down parser, little information is left around about why the sentence was not parsed. This is especially true if the unacceptable part occurs near the beginning of the sentence. (Our parser is working left to right.) A bottom-up parsing scheme has the advantage in this respect that constituents are recognized wherever they occur in the sentence. Combining a bottom-up parsing scheme with the semantic grammar provides a method for generating semantically meaningful feedback. After a sentence fails to parse, it can be passed through a bottom-up parser using a modified version of the semantic grammar.[6] Since the grammar is semantically based, the constituents found in the bottom-up parse represent "islands" of meaningful phrases. The modified semantic grammar can then be looked at to discover possible ways of combining these islands. If a good match is found between one of the rules in the grammar and some of the islands, another specialist can use the grammar to generate a response which indicates what other semantic parts are required for that rule. Even if no good matches are found, a positive statement can often be made which explains the set of possible ways the recognized structures could be understood. We think such positive feedback can be critical to breaking the user out of a vicious cycle of attempting syntactic paraphrases of a semantically unrecognizable idea by providing him explicit clues as to the set of things that can be understood by the system in that local context. Mechanisms for handling the conjunction and feedback problems as well as other issues relating to semantic grammars will be discussed in a later paper (Burton, 1975).

III. ON INFERENCING

In order to put the remaining part of the chapter in perspective let us review the different kinds of logical tasks that SOPHIE must perform. First there is the relatively straightforward task of answering hypothetical questions of the form: "If X then Y", where "X" is an assertion about some component or setting of the given instrument and "Y" is a question about its resultant behavior. A simple example might be: If the base-emitter junction of the

---

[6] Actually, a simplified, non-procedural form of the semantic grammar would be used. Here is a good example of using multiple representations of knowledge: a procedural (non-introspectable) version of the grammar for top-down parsing and a simplified non-procedural version used for making comments.

voltage limiting transistor opens, then what happens to the output voltage?"

The second task involves hypothesis evaluation of the form: "Given the measurements I have thus far made could the problem be X", where X is an assertion of the state of a given component. For example: "Could the base of Q3 be open?" What is at stake here is not determining whether the assertion X is true (i.e., whether Q3 is open in the faulted circuit!), but rather determining if the assertion X is logically consistent with the information already collected by the student. If it is not consistent, then the system must demonstrate why it is not. Likewise if it is consistent, the system must identify the subset of the collected information that supports the assertion and the subset which is independent of it.

The third logical task is that of hypothesis generation. In its simplest form this task involves constructing all possible "hypotheses" (or possible worlds) that are logically consistent with the known information, i.e., consistent with the information derivable from the current set of measurements. This task can be solved by the classical "generate and test" paradigm where the "test" part of the paradigm is performed by the previously mentioned hypothesis evaluation system. The "generate" part therefore forms the heart of this system.

The final task involves the complex and subtle issue of deciding whether a given measurement could in principle add any new information to what is already known. That is, is the given measurement logically redundant with respect to previous measurements or, stated differently, could the result of this measurement have been predicted from the previous measurements and a complete theory of the circuit.

We have found that all these logical tasks can be conveniently achieved by our model-driven, example-based, inference mechanism. Since simulation is at the heart of this system, we begin our technical discussion by considering how simulation is used to answer hypothetical questions.

A. Hypothetical Questions

A drastically simplified but intuitive view of one way to handle hypotheticals such as "If R11 shorts, then does the output current drop?" is simply to try it out and see. That is, instead of trying to deduce the consequences of R11 shorting, why not short R11 in some virtual but executable model of the given circuit, and then "run" the model to see what the consequences are? There are numerous complications to such a scheme, but instead of discussing them in the abstract, let us examine how this basic paradigm is realized in SOPHIE.

Given the hypothetical question "If X, then Y?", a procedural specialist, well versed in the inner workings of SOPHIEs general purpose simulator (Nagel & Pederson, 1973; Brown, Burton, & Bell, 1974), is passed the assertion X. This specialist first determines if the assertion unambiguously specifies a modification to the circuit.[7] If it does, then the specialist modifies the circuit description residing in the simulator so as to make it consistent with the assertion X.[8] Following this operation, the simulation model is executed, thereby producing as output a voltage table which specifies for each node in the circuit its voltage with respect to ground. This voltage table contains, either explicitly or implicitly, *all* the logical consequences of this modification under the current context or boundary conditions (i.e., instrument settings, load resistance, etc.).

Because this table contains a great deal of implicit information, it is treated as a structured data base by a collection of question-answering specialists which know how

---

[7] Examples of ambiguous or underspecified modifications are: capacitor being leaky -- how leaky; terminal being opened on a transistor -- what about the other terminals; beta shift in a transistor -- how much did it shift. In such cases as these, the specialist either queries the user or makes a default assignment, depending on the context of the question.

[8] This specialist enables (nontopological) modifications of the circuit to be made without requiring the simulator to redetermine the circuit equations. Hence, the invariant aspects of the circuit get analyzed once and compiled into an efficient model.

to derive information contained only implicitly in the data base. For example, a CURRENT specialist can determine from the data base the current flowing through every component or junction in the circuit; a RESISTANCE specialist can determine the "active" resistance of any component by using Ohm's law and the output of the CURRENT specialist. There is a power dissipation specialist and so on. The point is that each of these specialists has the knowledge (and inference capabilities) to compute additional (implicit) information contained in the generated data base.

A hypothetical question is then answered by transforming the question "Y" into calls to the appropriate question-answering specialists which construct the answer from the data base. Note the flow of information in answering the "if X, then Y" type question. X is first interpreted and "simulated" thereby generating a data base (or hypothetical world state) which implicitly contains all the consequences of X. Then Y gets interpreted, resulting in a directed action to infer particular information from this data base. Notice that the data base is generated without regard to Y -- a policy which on the surface may seem wasteful, but which proves not to be, as is discussed in Chapter 4. The above exposition has admittedly overlooked certain complications. For example, as anyone who has tinkered with any kind of complex systems well knows, a proposed modification to a system can result in disaster. Electronics is no exception. A modification often entails certain unexpected side effects of components sizzling into a vapor state which must somehow be captured in handling hypothetical questions. Stated somewhat more precisely, the data base or voltage table generated by the simulator satisfies only some of the constraints which constitute a complete theory of the circuit, its components and the general laws of electronics. In particular, the voltage table satisfies all of Kirchhoff's laws and the laws defining the behavior of transistors, etc. but the simulator does not attempt to satisfy "meta" constraints such as the limited power dissipation of the components. Indeed, since simulators are usually used to simulate working or near-working circuits there is little point in checking for such violations; but for our use such checks are crucial!

The simple paradigm mentioned above must therefore be expanded to handle the case where a proposed modification causes certain components to blow, i.e., a meta-constraint violation. Briefly, several new specialist are required. The first specialist examines the data base generated by the simulator in order to infer whether any meta-constraints such as power dissipation, voltage breakdown factor, etc. have been exceeded. This task involves repeated calls on the question-answering specialist. After determining all such violations it passes them on to another specialist which decides, using some heuristic knowledge of electronics, which violation is most severe and therefore which component is most likely to blow. This specialist then translates the selected violation (e.g., a particular resistor being overloaded) into a call for an additional modification of the circuit (e.g., that resistor opens) and fires up the model accordingly. Only one modification at a time is made since often one component blowing will "protect" another component even though initially both were overloaded. This process is then repeated until a point is then reached in which the output of the simulator satisfies all the meta-constraints.

The above process has now generated two important structures. First it has generated a kernel data base for various question-answering specialists. Second, it has generated a tree of possible fault propagations in conjunction with a control path of successive calls to the simulator. This latter structure reflects a sense of causality[9] which can be used to ascertain a causal chain of "important" events which followed from a particular modification.

There remains one crucial technicality worth discussing before we move on to the more novel uses of this inference scheme. This technicality has to do with the implicit quantifiers that usually lurk behind the scenes in nearly

---

[9] Since the simulator basically uses relaxation techniques, no local sense of causality is forthcoming from one particular simulation run; however, by factoring the "theory" of the circuit into constraints and meta-constraints we get the efficiency of relaxation but at the same time a sense of causality.

all hypothetical questions. For example, consider the hypothetical question: "If R22 shorts then does the output voltage change?" At first glance handling this question would seem straightforward: simply modify the circuit description (in the simulator) to make R22 have zero resistance and then execute the simulation system and examine the output voltage. Note, however, that nowhere has there been any specification of the boundary conditions (i.e., switch settings, load resistance, etc.) under which to run the simulator. In principle there can be an infinite number of conditions to try, each of which would require an execution of the simulator.

Our solution to this predicament involves the notion of using a weak or incomplete theory of the circuit to suggest potentially "useful" boundary conditions to be tried in order to obtain answers to particular questions. For example, SOPHIE first uses the present instrument settings -- remember such questions always occur in a context. Next, if the answer has not been determined (in this case, if output voltage has not changed) a specialist would attempt to construct a set of "critical" boundary conditions.

The heuristics underlying this specialist rely on the observation that all complex circuits have a hierarchical functional decomposition. Each module in this decomposition has both a structural description of its components (and their interconnections) and also a teleological description of its purpose in the overall design of the circuit or at least with respect to its superordinate module. From these teleological descriptions, it is possible to determine a set of boundary conditions which will force the circuit into a set of states which invoke or test out the various purposes of each module. Given these test cases[10]

---

[10] SOPHIE currently cannot deduce such cases. Instead, associated with the description of each module in the semantic net is an extensional specification of test cases derived from such qualitative knowledge as "when testing the Darlington amplifier make sure it can deliver its maximum current". Hence, when putting in a new circuit to SOPHIE, not only must a circuit description and a functional block description be put into the semantic net, but also the test cases for each functional block.

then all this specialist needs to do is to determine the chain of modules which contain the component that is being hypothetically changed and use these test cases as boundary conditions for the simulator. For example, in the above case, the specialist would discover that R22 was part of the Darlington amplifier (from accessing information in the semantic net) and that this module could be stressed by using a heavy load and setting the voltage and current controls to maximum. This *reasoning-by-example* paradigm is especially power-ful in SOPHIE since its use of simulation models provides a particularly effective technique for constructing or filling out examples that meet "interesting" conditions or constraints.

Before describing how this basic technique can be generalized to handle the spectrum of other logical tasks performed by SOPHIE, we call attention to the explicit factorization of processes and the multiple representations of knowledge underlying this single logical task. There are four basic modules to this factorization. The first is the simulator or data base generator which embodies a set of constraints reflecting general laws of electronics (e.g., Kirchhoff's laws, Ohm's laws), accurate models of transistors, resistors, capacitors plus a set of constraints defining the given circuit. Executing the simulation produces a description of a "world state" which simultaneously satisfies all these constraints. This data base constitutes the second module of this factorization. It, however, is not an arbitrary collection of assertions which describe the "world state" but is instead a carefully designed modelling structure embodying a kernel set of information from which the answers to any questions about the state can be efficiently derived through the application of question-answering or measurement specialists. These inference specialists constitute the third module. They, likewise, embody general principles of electronics and use these principles (much as Consequent theorems are used in Planner) to determine information that is contained only implicitly in the data base. Since the generated data base is always of a specific form, the question-answering specialists can be designed to take advantage of this invariant structure by having all their "how to do it" type knowledge encoded in terms of how to operate on this fixed set of kernel relations or predicates. The significance of this three-part factorization is discussed at greater length in Chapter 4.

The fourth module contains qualitative knowledge (e.g., what components are most likely to blow, how power amplifiers can be stressed) and heuristic strategies for combining the qualitative knowledge with the other three modules. Speaking somewhat metaphorically, this fourth module may be viewed as a "weak" or incomplete theory (of electronics) which has been constructed for carrying out a particular task. Much of what follows concerns how additional weak theories can be used to augment the powerful but narrow capabilities of the data base generator (and its corresponding interrogators) so that it can be used to perform other kinds of logical tasks besides just answering hypothetical questions.

B. Hypothesis Evaluation

Hypothesis evaluation is the process of determining the logical consistency of a given hypothesis[11] with respect to the information derivable from the current set of measurements. It is important to realize that a hypothesis can be logically consistent with the known information and still not be correct in the sense of specifying what is actually wrong with the circuit. For example, if no measurements have been performed -- meaning that in principle no information is known about the behavior of the instrument -- then many hypotheses are acceptable (i.e., those which are syntactically consistent). If, however, some measurements have been made, then the task for the hypothesis evaluation specialist is to partition these measurements into three classes. One class contains the measurements that are contradicted by the given hypothesis, another class contains the measurements which are logically entailed by the hypothesis and the last class contains those

---

[11] A hypothesis concerns the state of a given component such as a capacitor being shorted, a resistor being open, a transistor being shorted, etc.

measurements that are independent of any of the logical consequences of the given hypothesis.

Although these partitions are only over the set of measurements the student has taken, they are determined by taking into consideration all the logical implications derivable from the given hypothesis. If, for example, a hypothesis concerns a particular component being shorted, there need be no direct or obvious measurement on that component for that hypothesis to be either supported or refuted! By taking into consideration both the local and global interactions of components in the circuit, measurements arbitrarily far away from the hypothetically faulted component may be used to support or refute the hypothesis.

By restricting the domain of acceptable hypotheses to statements specifying faulty components of the circuit, simulation can be used to determine the consequences of a hypothesis much as it was used to infer the consequences of the assertional part of a hypothetical question. Unlike the handling of hypothetical questions, there is no inherent problem with determining which boundary conditions to use. We simply use the same set of boundary conditions which the student used while performing his given set of measurements. (Each measurement has associated with it the complete specification of how the instrument was set up when that measurement was taken.)

To facilitate a concise description of how a student's hypothesis is evaluated, we introduce the notion of a context frame which consists of the set of measurements the student made under one particular setting of the instrument. In other words, a context frame is all those measurements made under the same boundary conditions. The hypotheses evaluation specialist proceeds as follows: First it selects a context frame (using various psychological considerations such as recency, number of measurements, etc). It then uses the boundary conditions of that context to set up a simulation of the hypothesis. The output of the simulation is then used by the question-answering or measurement specialists to reconstruct, in this generated "hypothetical world", all the measurements composing the selected context frame. If the values of any of these measurements are not equivalent to the ones taken by the student in the given frame, then a counterexample or inconsistency has been established. Depending on other considerations (of either a pedagogical or logical nature) another context frame is selected and the process is repeated. If none of the context frames yields a contradiction, then the hypothesis is accepted as being logically consistent with all the known information.

There remain two unresolved issues. First we have not specified how to separate those measurements supported by the hypothesis from those which are independent of it. Second, and much more important, we have relied exclusively on the quantitative replication of the values of these measurements in the hypothetical world (i.e., the world entailed by the hypothesis) with those actually obtained by the student. This is a most precarious strategy, for few people can construct hypotheses that exactly mimic the quantitative behavior they have thus far observed, and furthermore, there is no reason why they should be able to! What is reasonable to expect is a more qualitative, common sense mimicry of the results in the observed world by those in the hypothetical world. In order to determine this, a "metric" is used to decide if the two exact quantitative values of a measurement (each performed in its "world") are *qualitatively* similar. For example are .3 and .9 "equivalent" values for the voltage at some node? In principle, this metric must incorporate both a general theory of electronics (such as the expected voltage range of a forward biased base-emitter junction) plus a structural theory of the particular circuit.

Our solution to this problem employs a heuristic which circumvents much (but not all) of the need for employing such theories. It is based on the observation that the value of a given measurement in a working circuit can be used to qualitatively normalize the distance between the two values of that measurement obtained in the hypothetical and observed "worlds". If the hypothetical and observed values are split by the value obtained in a working circuit, then the distance between the hypothetical and observed value is qualitatively large and therefore constitutes a counterexample to the given hypothesis.[12] If, however, the value obtained in the working circuit does not split the hypothetical and observed values, then the distance between them is a simple function (conditioned product) of (i) how far apart the two values are (their percentage difference) and (ii) the minimum of the differences between each of them and the working circuit value.

Assuming that a given measurement is not contradicted by the hypothesis, there is the issue of deciding if it actually supports the hypothesis or is just independent of it. This decision is reached by seeing if the value of that measurement in the correctly functioning circuit is qualitatively similar to values obtained in the hypothetical and observed "worlds". If it is, then the given measurement does not reflect any symptoms of either the actual fault or the hypothetical fault and is therefore independent of the hypothesis; but, if the value in the working circuit qualitatively differs from the other two similar values, then that measurement supports the hypothesis.

C. Hypothesis Generation (Theory Formation)

One of the more difficult logical tasks performed by SOPHIE is determining the set of possible faults or hypotheses that are consistent with the observed behavior of the faulted instrument (i.e., all the measurements taken up to that time). Such a capability is useful for several reasons. First, it can be called on by a student either when he has run out of ideas as to what could be wrong (i.e., what faults have not yet been ruled out by his measurements) or when he wishes to understand the full implications of his last measurement. It can also be called on by a tutorial specialist which might use this facility to detect the subtle ramifications of a measurement just performed by the student and thereby decide to query him

---

[12] This is the case unless the absolute difference between these measurements is less than some given threshold.

as to its significance. In a somewhat similar fashion, this facility already plays a major role in judging the quality of a given measurement (as will be discussed later) and in principle could be used to troubleshoot the instrument automatically.

The method of constructing the set of hypotheses uses the venerable "generate and test" paradigm: first, a backward working specialist, called the PROPOSER, examines the value observed for an external measurement and, from that observation, determines a list of all possible significant hypotheses which more or less explain that one measurement. This specialist uses a procedural form of simple production rules to encode its limited knowledge. Because the PROPOSER is not endowed with enough knowledge to capture all the complex interactions and subtleties of the circuit, it often errs by including a hypothesis that does not explain the observed behavior. In other words, the list it produces is overgeneral.

It is then the job of another specialist called the REFINER to take this overgeneral list and refine it. The REFINER, in essence, "simulates" each fault on the PROPOSERs list to make sure that it not only explains the output voltage (as a check on the PROPOSER) but also that it explains all the other measurements that the student has taken. By having the REFINER simulate each hypothesis, it takes into consideration all the complex interactions that a linear theory of the circuit fails to capture.

Counting on the simulator to check out all the subtle consequences of a proposed "theory" or hypothesis, however, leads to one major problem. For the REFINER to be able to simulate a hypothesis, the hypothesis must specify an explicit fault or modification to the circuit; but often the PROPOSER (like people) generates a *fault schema* which represents an infinite but structurally similar class of faults. For example, the hypothesis "the beta of the Darlington amplifier (of the IP-28) is low" is one such fault schema as is the hypothesis "C2 is leaky". For the first hypothesis it is not clear what the proposed beta is, just that it is lower than it should be, and for the second hypothesis, it is not specified how leaky C2 is or what its leakage resistance is supposed to be. In other words, a fault schema is an underspecified fault which has at least one unspecified parameter in its schema definition.

It is the job of another specialist, called the INSTANTIATOR, to take a fault schema and fill out or instantiate the unspecified parameters as best it can using an incomplete theory of the circuit. Once these parameters are instantiated, the fully specified hypothesis must then be checked to see if it really accounts for all the known measurements. For example, a subtle situation can arise where given any one context frame of measurements the proposed fault schema can be instantiated so as to be consistent with all the information derivable from that context frame. If, however, we simultaneously consider two or more context frames, we might discover there exists no consistent instantiation of the schema (i.e., the instantiated value created for one frame does not equal the value created for the second frame).

The INSTANTIATOR uses several techniques to determine a potentially consistent specification of a fault schema, the most general of which is a simple hill climbing strategy in which a specific value for the fault schema is guessed and then partially simulated (that is, the output voltage is determined). From the result of that simulation another guess is made until finally a value is found that causes the desired behavior in the given context frame.

As must be apparent, the hypothesis generator's numerous calls to the "simulator" could cause SOPHIE to consume countless cpu minutes before generating a set of viable theories. To avoid this, the REFINER and INSTANTIATOR use, in addition to the full-blown circuit simulator, a hierarchical, functional-block simulator. This latter simulator can either execute a functional block in the context of the whole circuit or simply in isolation as happens when the INSTANTIATOR wants to determine only the local effects of an instantiated fault schema. By correctly coordinating and maintaining consistency between these multiple representations of the circuit, several orders of magnitude of speed (over using just one simulator) can be realized.

Before leaving this section, it is worth noting that the INSTANTIATOR can be used in quite subtle ways to rule out certain faults even when no specific symptom has yet been encountered. For example, suppose the correct output voltage has been determined under a given load. The INSTANTIATOR could use this fact to determine a *range of* possible values for a given fault schema such as effective beta of the Darlington amplifier (i.e., it can determine a lower bound of the combined beta such that if the beta had been any lower, the output voltage would have dropped and hence been symptomatic). Suppose then that the lower bound of this beta range was greater than betas of each of the two transistor making up the Darlington. This fact, in turn, would imply that neither of these transistors could be shorted! (Note then under many situations, one of these transistors could be shorted without there being any external symptoms.)

D. Determining Useless Measurements

Perhaps the most complex logical and tutorial tasks performed by SOPHIE and also one which illustrates a novel use of hypothesis generation is the task of verifying whether or not a given measurement *could* possibly add any new information to what is already known. If, for example, the result of a given measurement could be logically deduced from the previous measurements (using all the axioms of electronics, a complete description of the circuit, and a description of all the inputs and all possible faults, etc.), then this measurement could add no new information. In other words, this measurement would be logically redundant with respect to the prior measurements.

A moment's consideration reveals the potential complexity of proving that a measurement is redundant. Such a proof must take into consideration not only all the structural properties of each module making up the instrument but the functional relations between that module and all other modules as well. That is, it must take into account the global "purpose" of each component and each module in the overall design of the circuit!

Our approach to handling this task stems from an analogy to how one might prove the independence of an axiom set. If one is trying to establish that a new axiom is independent from a given collection of axioms, one might try rather to deny it by constructing a world model (or said differently, a "possible world") for the original axiom set which is not a world model for the augmented axiom set. Likewise one could consider the set of all "world models" or possible worlds for the original axiom set and determine if this set is "reduced" by adding the new axiom to the original set.

In our case, we view each measurement (along with its value) as an axiom or assertion and proceed according to the above analogy. The set of hypotheses constructed by the hypothesis generator serves as the set of all possible "world models" which satisfy or are consistent with the known measurements. Now a new measurement is taken and its value likewise passed to the hypothesis generator. If the resultant set of possible worlds is a proper subset of the prior set, then this measurement has added new information by eliminating at least one of the possible worlds (i.e., faults) and is therefore not redundant. Likewise if the set of possible worlds is the same (i.e., before and after the measurement was taken), this last measurement has added no information and is therefore logically redundant.[13]

E. Combining Qualitative and Quantitative Models of Knowledge

One of the more exciting possibilities for expanding SOPHIEs capabilities is in the interfacing of rule-based qualitative models of knowledge with SOPHIEs more quantitative models. Although there are many good reasons

---

[13] The above argument is, of course, not a proof and is only intended to be suggestive. In fact, since the hypothesis generator constructs only single fault possible "worlds" the above analogy is not literally true. Nevertheless, if the user is told that the circuit has only a single fault, then his space of possible worlds coincides with the hypothesis generator's space. The argument also becomes much more complex if the circuit has memory. In general, it must be shown that there cannot exist two measurements which taken individually do not rule out a possible world but taken collectively, do.

for investigating the interplay of qualitative and quantitative models of knowledge (deKleer, 1975), ours is driven by the awareness that our basic inferencing scheme has a major drawback. Namely, it achieves most of its answers without being able to generate a description of why the answers are true (i.e., a proof) or how the answer was derived. For example, SOPHIE can decide if a measurement is redundant, but it cannot "explain" or justify its decision in terms of causal reasoning. Indeed, part of the efficiency of our reasoning paradigm stems precisely from this fact. (See Chapter 4 for a theoretical discussion of this point.)

As a result we are beginning to investigate ways to combine an incomplete but qualitative (rule-oriented) type inference system with the "complete" model-driven schemes detailed in this chapter. In particular, we are intrigued with the possibility of using an "incomplete" qualitative theory to create a rationalization for an answer derived by the quantitative model-driven scheme. For example, once SOPHIEs hypothesis evaluator identifies which measurements contradict a given hypothesis, it seems like a much easier task to then "explain" why these measurements are counterexamples. Likewise, in handling hypothetical questions we do not need to count on the qualitative rules to sort out what does happen from what might plausibly happen. The quantitative models can do that.[14] Once we know for sure what happens, however, we can then use the qualitative rules for generating plausible causal explanations. Note that such an explanation can be useful even if it is not logically complete. It just has to highlight certain steps in the causal chain of reasoning.

Another possible direction in which to investigate combining quantitative and qualitative models of knowledge is to construct qualitative models for handling the ac and transient aspects of a circuit. In particular, the dc-based

---

[14] Note that actually we have here the chance to combine two completely different uses of qualitative reasoning. The first kind is used to create the interesting "examples" which are then passed off to the simulator. Once the simulator does its thing the second kind of qualitative reasoning could be called on to help explain the answer!

quantitative models can be used to determine the operating points of transistors. This information can then be used by qualitative ac specialists to predict such properties as clipping, distortion, etc. Likewise, a qualitative model could call a quantitative model to resolve any encountered ambiguities such as a feedback situation in which it is not clear, from a purely qualitative point of view, which of two opposing forces actually wins. In fact some of the original ideas for SOPHIE grew out of wanting to make such extensions to our purely qualitative reasoning scheme which used augmented finite state automata to model the qualitative properties and interactions of processes (Brown, Burton, & Zdybel, 1973).

IV. CONCLUSION

SOPHIE is sufficiently operational that it is ready for experimental use in a realistic instructional environment. Although it is a large and complex system, it is surprisingly fast, yielding response times in the order of a few seconds on a lightly loaded TENEX and requiring typically two cpu minutes per hour of use. We believe that much of this efficiency is due to: (i) the use of multiple representations of the constant or universal portions of SOPHIEs knowledge; (ii) the use of simulation as a general synthesis procedure for generating a world state description (or data base) which satisfies a given set of constraints; (iii) the close coupling of the structure of the world state description with the analysis or question-answering procedures that operate on it; and (iv) the use of heuristic strategies for expanding the domain of applicability of this highly tuned example-based reasoning paradigm so that it can handle the complex tasks of hypothesis evaluation, hypothesis generation, and redundancy checking. The ability of this scheme to handle complex, multistate feedback systems is encouraging since it is precisely these kinds of "worlds" that are most difficult to capture within the classical AI paradigms.

ACKNOWLEDGMENTS

We are indebted to Alan G. Bell for his extensive help in building the simulators and the hypothesis generation system and to Rusty Bobrow for the countless hours he has spent discussing the theoretical ramifications of SOPHIE. Much of our current view of SOPHIE stems from his observations. This work was supported in part by the Advanced Technical Training program ARPA-HRRO, and Air Force Human Resources Laboratory Technical Training.

REFERENCES

Brown, J. S., Burton, R. R., & Bell, A. G. SOPHIE: A step towards a reactive learning environment. *International Journal of Man-Machine Studies*, in press, 1975.

Brown, J. S., Burton, R. R., & Bell, A. G. SOPHIE: A sophisticated instructional environment for teaching electronic troubleshooting (an example of AI in CAI). (BBN Report No. 2790). Cambridge, Mass.: Bolt Beranek and Newman, Inc., March, 1974.

Brown, J. S., Burton, R. R., & Zdybel, F. A model-driven question answering system for mixed-initiative computer assisted instruction. *IEEE Transactions on Systems, Man and Cybernetics*, May, 1973, *SMC-3*.

Burton, R. R. Semantically centered parsing. Ph.D. Dissertation, University of California at Irvine, December, 1975.

Carbonell, J. R. AI in CAI: an artificial intelligence approach to computer-assisted instruction. *IEEE Transactions on Man-Machine Systems*, December, 1970, *MMS-11* 190-202.

deKleer, J. Qualitative and quantitative knowledge in classical mechanics. Working Paper, MIT-AI Laboratory, February, 1975.

Nagel, L. W., & Pederson, D. O. SPICE: simulation program with integrated circuit emphasis. Memorandum ERL-M382, Electronics Research Laboratory, University of California at Berkeley, April, 1973.

Teitelman, W. INTERLISP reference manual. Xerox Palo Alto Research Center, October, 1974.

Winograd, T. *Understanding natural language*. New York: Academic Press, 1972.

Woods, W. A. Transition network grammars for natural language analysis. *Communications of the Association for Computing Machinery*, 1970, *13*(10), 591-606.