A PROBLEM SOLVING APPROACH TO GENERATING TEXT FROM SYSTEMIC GRAMMARS*

Terry Patten

Dept. of Artificial Intelligence, University of Edinburgh
Hope Park Square, Meadow Lane, EH8 9NW.

ABSTRACT

Systemic grammar has been used for AI text generation work in the past, but the implementations have tended be ad hoc or inefficient. This paper presents an approach to systemic text generation where AI problem solving techniques are applied directly to an unadulterated systemic grammar. This approach is made possible by a special relationship between systemic grammar and problem solving: both are organized primarily as choosing from alternatives. The result is simple, efficient text generation firmly based in a linguistic theory.

INTRODUCTION

This paper will describe an approach to text generation where AI problem solving techniques are used to generate text from systemic grammars.** Problem solving is a general term used here to refer to areas of AI research such as 'expert systems', 'planning', 'design' and so on (Hayes-Roth et al., 1983). Techniques developed in these fields are applied directly to an unadulterated systemic grammar, resulting in a simple, efficient text generator firmly based in an established linguistic theory.

This approach is only possible because of a fundamental relationship that exists between systemic grammar and AI problem solving. This relationship is described in the next section. The third section will be concerned with one of the most important manifestations of this special relationship: a common representation. The following section will show how this common representation allows goal directed problem solving techniques to be applied directly to the grammar. One of the most novel and important aspects of this approach is that it is compatible with the semantic stratum described in the systemic theory: a system network organized around the idea of 'register' (Halliday, 1978). The semantic stratum and its relationship to the grammar will be discussed next. Some advantages of the approach will then be put forward. Finally, the current status of the project will be described, including sample output generated from a large grammar.

THE FUNDAMENTAL RELATIONSHIP

> 1. "The central nature of intelligent problem solving is that a system must construct its solution selectively and efficiently from a space of alternatives." (Hayes-Roth et al., 1983)
>
> 2. "We shall define language as 'meaning potential': that is as sets of options or alternatives, in meaning, that are available to the speaker-hearer." (Halliday in deJoia et al., 1980, #572)

Compare these two quotations. Notice that both AI problem solving and systemic grammar have at their very core the idea of choosing from alternatives. Systemic grammar is probably unique in having such emphasis on the idea of choice; or in different terminology, systemic grammar is distinguished in allowing the paradigmatic mode of description to dominate over the syntagmatic (see Halliday et al., 1981, p. 19). Thus, this is a special relationship between systemic grammar and AI problem solving.

This fundamental relationship can be exploited because systemic grammar provides knowledge (in the AI sense) about the various linguistic alternatives, and AI problem solving provides computational techniques for choosing between the alternatives, given the appropriate knowledge. The text generation approach described here is simply the standard AI knowledge-based problem solving methodology, with a systemic grammar acting as part of the knowledge base.

KNOWLEDGE REPRESENTATION

One of the manifestations of this fundamental relationship between AI problem solving and systemic grammar is a common representation of knowledge. Both of these fields represent the interdependencies between the various alternatives as "condition/effect" relationships.

* Many thanks to my supervisors Graeme Ritchie and Austin Tate. This work was supported in part by an ORS award.

** For an overview of systemic grammar, see (Winograd, 1983) Chapter 6.

problem solving

The last decade has produced problem solving techniques which use domain-specific knowledge to guide the problem solving process. Problem solving knowledge is often expressed as condition/effect rules. For instance, a medical problem solver may have the rule:

if
    a patient has symptoms X, and Y
then
    drug A should be administered.

Here if the conditions (the symptoms) are satisfied, the problem solver can infer that drug A should be given. At this point other rules may be involved:

if
    a drug should be administered and it is not in stock
then...

The problem solver is forming a chain of inferences which leads toward the solution. This is called "forward chaining".

Condition/effect rules can also be used to reason from the effects back to the conditions. Suppose we have a rule:

if
    a surface is hot and compound B is applied
then
    the surface will be made permanently non-reflective.

If a problem solver has a goal to make a surface non-reflective, it can see from the effects that this rule will achieve the goal. The conditions of the rule are set as subgoals, and the problem solver will try to find rules to achieve these. Rules must then be found that have the effects of heating the surface and applying the compound. Here the problem solver is working backward from the solution. This is called "goal-directed backward chaining".

systemic grammar

Systemic linguistics was developed in the early sixties by M.A.K. Halliday, although its roots in sociology and anthropology extend back much further. The emphasis of systemic linguistics has not been on the structure of language, but on its function; systemicists are not so much interested in what language 'looks like', as in how it is used. They are interested in how language is used to ask questions and make statements, how language can be used to relate 'who did what to whom', and how language ties itself to previous discourse.

The relationship between this functional view of language and the structural tradition is analogous to the relationship between physiology and anatomy*, and is equally complementary. This functional perspective has led to a different conceptualization of what language is, and how it should be described.

The most important knowledge structure in systemic grammar is the 'system' — this is where the theory gets its name. A system is simply a mutually exclusive choice between a set of alternative features. Figure 1 shows a system that represents a choice between a marked- and unmarked-wh-theme.

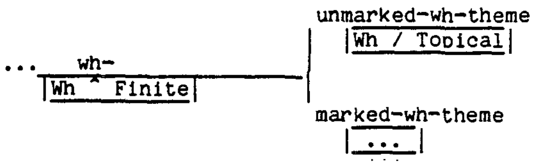

Figure 1. A system (Mann/Halliday).

Systems also have 'entry conditions': a logical combination of features that must be chosen before the particular choice is appropriate. In this case the entry condition is simply the feature wh-. So the clause must be a wh- clause before the choice between a marked- or unmarked-wh-theme is relevant. The boxes contain what are called 'realization rules'. These specify the syntactic consequences of choosing the associated feature. "Wh / Topical" is read: "the Wh element is conflated with the Topical", meaning that the Wh and Topical are realized by the same item. "Wh ^ Finite" is read: "the Wh element is adjacent to the Finite element", meaning that the Wh element immediately precedes the Finite element in the clause.

As well as systems, systemic grammars may contain what Mann (1983) calls 'gates'. A gate also has some logical combination of features acting as entry conditions.

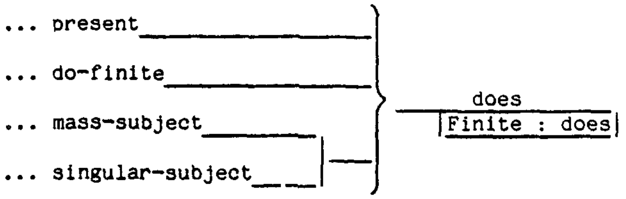

Figure 2. A gate (Mann/Halliday).

In Figure 2 the curly bracket means AND, and the square bracket means OR. A gate also may have realization rules. Here the Finite element is constrained to be some form of 'does': "does", "does not" or "doesn't". The significant difference between systems and gates is that gates do not involve a choice.

* This analogy was probably first made by Firth (1957) and has been used several times since — see (Winograd, 1983, p.287)

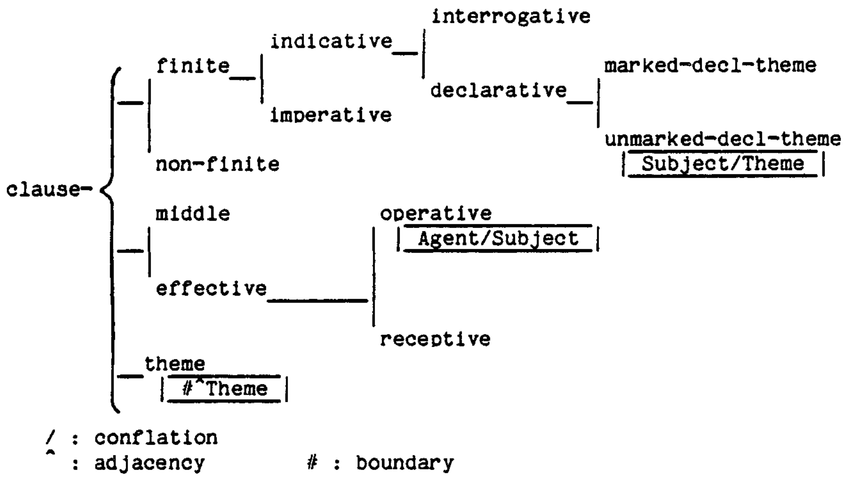

Figure 3. A grammar excerpt.

Now consider these two constructs from a problem solving point of view. A feature that is part of a system can be interpreted as a condition/effect rule. The conditions are simply the entry conditions of the system; the effects are choosing the feature, and doing whatever the realization rules say. This means that these features can be interpreted as problem solving rules and put at the disposal of the problem solver. Again it must be stressed that a system involves choice. From a problem solving point of view choices should be avoided whenever possible, in case the wrong choice is made. Notice if a system feature is used for backward chaining the choice is not explicitly considered. Suppose there is a goal to choose unmarked-wh-theme. Since the problem solver can interpret the system features as condition/effect rules, it sees that there is a rule called unmarked-wh-theme that achieves this goal as one of its effects. The problem solver begins to backward chain by invoking this rule and setting its condition, wh-, as a subgoal. The feature marked-wh-theme was never explicitly considered.

Similarly, features that are gates can be interpreted as forward chaining condition/effect rules. In Figure 2, if the entry conditions are satisfied, the does rule fires, choosing does and constraining the Finite element.

THE METHOD

The last section showed that features from systemic grammars can be interpreted as a condition/effect rule of the type used by AI problem solvers, regardless of whether they are part of a system or a gate. An AI problem solver can thus use a systemic grammar as part of its knowledge base, and solve grammatical problems in exactly the same way as it solves medical problems using medical knowledge, or chemistry problems using chemistry knowledge.

an example

Figure 3 is a simplified excerpt from a systemic grammar. Suppose, for the moment, that the semantics wants to choose unmarked-declarative-theme and operative. The grammar provides rules that achieve these goals as part of their effects. The feature unmarked-declarative-theme can be thought of as a rule that chooses that feature and conflates the Subject with the Theme. This rule has, however, the condition declarative. This is set as a subgoal which can be achieved by another rule that in turn has the condition indicative. In this way the problem solver backward chains from unmarked-declarative-theme through declarative, through indicative, through finite, to clause. At this point the backward chaining stops because clause has no conditions. The problem solver also backward chains from operative through effective to clause. Once clause is chosen, the gate theme fires (the only instance of forward chaining in this example).

Every time a rule is used the 'realization rules' in the effects are accumulated, gradually constraining the structure of the clause. In the example, the Agent has been constrained to be the leftmost constituent in the clause. The semantics will choose other features of course, from parts of the grammar not shown here, and after further forward and backward chaining, the clause will be completely determined.

The careful reader may have noticed that it is possible for the semantics to start the same process with the goal "move the agent into the theme position" (conflate Agent and Theme), assuming there is a rule expressing the transitivity of conflation. The transitivity rule would set as subgoals: "conflate Agent with X" and "conflate Theme with X", where X could be instantiated to Subject. From there the problem solving would proceed as before. However, this would require far too much inference for such a simple goal. First, the transitivity would have to be worked out correctly. Second, there are likely to be other rules with the same realization rules, but which would lead to conflicts, and backtracking.

In problem solving, if a simple goal requires too much inference, its solution can be 'compiled' (Brachman, 1983). Here, the semantics may have a rule that says:

```
if
    there is a goal to make a statement and a
    goal to move the agent into the theme
    position
then
    choose unmarked-declarative-theme and
    operative.
```

This use of compiled knowledge to actually choose features from the grammar corresponds to the systemic idea of 'preselection'. Preselection is an important part of systemic theory, being the vehicle of realization across network boundaries. Systemic grammar adopts

> ... the general perspective on the linguistic system you find in Hjelmslev, in the Prague school, with Firth in the London school, with Lamb and to a certain extent with Pike - language as a basically tristratal system: semantics, grammar, phonology. (Halliday, 1978, p.39)

Each level must pass down information to the level below. Realization rules at the higher level preselect features from the next level below. The semantic stratum (described in the next section) preselects features from the grammatical stratum (e.g. unmarked-declarative-theme and operative in the example above). Similarly, the grammatical stratum preselects phonological/graphological features.

Preselection is also used to interface the different ranks at the grammatical level (clause, group and word). The colon in Figure 2 is the symbol for preselection. Thus the feature does at the clause rank preselects the feature does from the auxiliary network at the word rank. If, for instance, the features reduced and negative are also preselected, the Finite element will be realized as "doesn't".

Returning to Figure 3, compare this backward chaining approach to Mann's (1983) NIGEL system. NIGEL begins at the left hand side of the network and works its way towards the right. It starts by choosing the feature clause. Then it sees that it must choose between finite and non-finite. There is a semantic 'choice-expert' associated with this system which cannot make the choice without specific information about the context and the communicative goals. The choice expert gains this information by passing messages to the 'environment'. In this case the answer returned from the environment will indicate that finite should be chosen. Another choice expert will now choose between indicative and imperative and so on.

Whether or not this is a valid or interesting way to do text generation is not at issue here. From a computational point of view NIGEL has some drawbacks. Most importantly, an explicit choice must be made for every system encountered during the process. For large grammars, this will number in the hundreds, and will result in a large overhead. In contrast, the preselection - backward chaining approach outlined in this paper greatly reduces the number of explicit choices.

The reason these choices are avoided here is that the problem solving process is goal-directed. The semantic stratum chooses some features from the right hand side of the network, which greatly reduces the number of possible paths through the network from the very start.

It could be argued that this kind of goal-directed search is non-deterministic because systems may have disjunctive entry conditions. There is, however, an AI problem solving technique which has been developed for this purpose: least commitment (Stefik et al., 1983). Least commitment is simply the principle of not making any choices until absolutely necessary. Whenever a disjunctive entry condition is encountered, a decision must be made about which subgoal to set. There is no requirement that the decision be made at that particular instant, so it is suspended until one of the subgoals is set as part of another chain in inference (gratuitously solving the original problem). Of course there will be cases where none of the subgoals (entry conditions) are part of another inference. In these cases, it must be assumed that the semantics will preselect a feature corresponding to one of the subgoals.

Clearly this whole text generation method relies on the semantic level to preselect the appropriate grammatical features. The next section will briefly look at this semantic level.

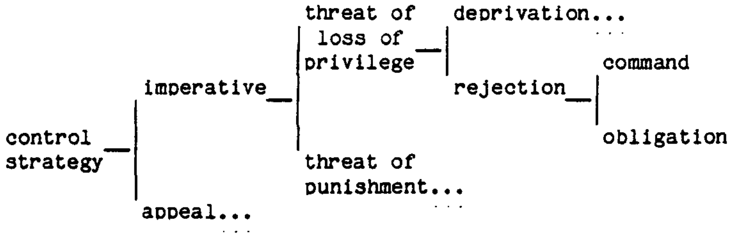

Figure 4. Some semantic choices

SEMANTICS

No motivation for the stratified approach adopted by systemic grammar will be given here, except pointing out that the role of the semantic stratum is to interface the extra-linguistic with the grammatical (Halliday, 1978). In order to preselect the correct features from the grammar, this level must contain a considerable amount of knowledge (in the AI sense) relating grammatical features to extra-linguistic factors.

In this section we will look at one particular organization of the semantic stratum, as presented in (Halliday, 1978). Halliday organized his semantic stratum around the idea of 'register':

> It refers to the fact that the language we speak or write varies according to the type of situation ... What the theory of register does is attempt to uncover the general principles which govern this variation, so that we can begin to understand what situational factors determine what linguistic features. (Halliday in deJoia et al., 1980, #764)

Halliday uses the same system network notation to describe the semantics. Figure 4 (adapted from (Halliday, 1978)) describes the control strategies that a mother can use on her child.

The features of a semantic system network, like those of the grammatical networks, have realization rules — including preselection. For instance the semantic feature rejection preselects the features which will make the hearer the Medium (Affected), and realize it with the pronoun 'you' (by preselecting from the nominal group and noun networks). The semantic feature decision preselects, for instance, the clause feature declarative. The semantic feature resolution preselects the features present-in and present to give this type of threat its tense construction — e.g. "you're going upstairs", "I'm taking you upstairs". Similarly, obligation preselects necessary passive modulation (Halliday, 1970) — e.g. "I'll have to take you upstairs", "you'll have to go upstairs" (Halliday, 1978).

Unfortunately, very little work has been done in the area of register, even by Halliday and his colleagues, so no large portions of a semantic stratum have been built. However, this example illustrates the idea.

ADVANTAGES

The backward chaining approach outlined here has several advantages. First, this method does not involve any linguistic sacrifices, since an established linguistic formalism is utilized. Systemic grammar was developed by linguists for linguistic purposes, and is used here in a totally unadulterated form. Nothing linguistically ad hoc has been introduced for computational reasons.

Second, no computational sacrifices have been made to accommodate the linguistic formalism. State-of-the-art computational techniques are being exploited at all stages of the problem solving process.

Third, the approach is parsimonious. There is no need for a special-purpose text generation component. Other methods involve an AI problem solver that does the extra-linguistic work and perhaps the high-level 'text-planning', then passes a specification off to a special-purpose mechanism that processes the grammar. Here the AI problem solver can directly process the grammar, eliminating the special purpose component, and avoiding the kind of message passing that NIGEL, for example, must do.

PROJECT STATUS

At present, this approach to text generation is being tested on a large systemic grammar. The grammar has been collected from a variety of sources (Mann/Halliday) (Kress, 1976) (Halliday & Hasan, 1976) (Winograd, 1983), and contains about six hundred grammatical features. Fragments of grammar usually appear in the linguistic literature as 'system networks'. These are entered as LISP data structures, and translated by a three page LISP program into OPS5 production rules. (OPS5 is a widely used production system that was used to implement, for example, R1 (Gaschnig et al., 1983)).

Once the grammar is in the form of OPS5 rules, OPS5 can perform forward and backward chaining directly. The rest of the system consists mostly of OPS5 rules to act on the realization rules of the grammar, and to output the text as it is being generated.

The interface between the grammar and the semantics has been implemented, namely preselection. Since preselection is done via realization rules, it is implemented by a small group of OPS5 rules as just mentioned.

Although the interface between the grammar and the semantics has been implemented, the semantic stratum itself has not. This means that to test the approach, those features that would have been preselected by the semantics must be preselected by hand.

Another limitation at the moment is that there is no graphological level. This means that the output does not contain punctuation, capitals, the word "an", and so on.

To put all this in perspective, recall that systemic linguistics stratifies language into the semantic, the grammatical, and the graphological (or if working with speech, phonological) strata. Currently only the middle stratum, the grammatical; has been implemented. Again it should be pointed out that the interface between the different strata (preselection in each case) has been implemented as well.

sample output

Consider the context of a medical expert system that is trying to diagnose a patient's illness. Suppose there is a patient named Mary who has been having headaches and stiff neck muscles. The expert system hypothesizes that Mary has a fever, and tests this hypothesis by asking "Does Mary have a fever ?" At this point, the user, who we will assume is neither a medical or computing expert, can ask "WHY" (did you ask me that question?)*. The test system at this stage can generate the following response (bars have been added to indicate clause boundaries).

```
|| well mary has been having headaches ||
on this basis perhaps she has a infection
|| this possibility would be supported by
a  fever || so we ask | does she have one
||
```

Remember that at present, the features that would be preselected by the semantics must be preselected by hand for each individual clause. However, this example illustrates the grammar we are working with, and demonstrates that this approach works very well with large grammars.

CONCLUSION

This paper has described a new approach to generating text from systemic grammars. State-of-the-art AI problem solving techniques are applied directly to an unadulterated systemic grammar. We have seen how this approach is made possible by a special relationship between systemic linguistics and AI problem solving. A semantic stratum, consisting of a large knowledge base relating different 'registers' to grammatical features, preselects some features from the grammatical level. The large number of features which are not preselected are inferred efficiently by goal-directed backward chaining and forward chaining.

This approach has the advantage of being able to combine an established linguistic formalism with powerful AI methods. It also has the advantage of simplicity resulting from the application of these same methods throughout the generation process.

This approach has been applied successfully to a large grammatical stratum. Of course it will not have been tested properly until a substantial semantic stratum is developed.

In conclusion, although there are still many unresolved linguistic matters in systemic text generation, we hope this approach has moved toward solving the computational problems involved.

* Following an example from (Hasling et al., 1984).

REFERENCES

Brachman,R., Amarel,S., Engelman,C., Engelmore,R., Feigenbaum,E., Wilkins,D. "What are Expert Systems ?" In (Hayes-Roth et al., 1983).

Firth,J.R., "A synopsis of linguistic theory 1930-1955)." Studies in Linguistic Analysis. Blackwell, Oxford, 1957, pp. 1-32. Reprinted in Palmer, F.R.,(ed.) Selected Papers of J.R.Firth 1952-1959. Longman, London, 1968, pp. 168-205.

Forgey,C.L. "OPS5 User's Manual". CMU-CS-81-135 Carnegie Mellon University, Pittsburgh, 1981.

Gaschnig,J., Klahr,P., Pople,H., Shortliffe,E., Terry,A., "Evaluation of Expert Systems: Issues and Case Studies." In (Hayes-Roth et al., 1983).

Halliday, M.A.K., Explorations in the Functions of Language. Edward Arnold, London, 1973.

---, Language as a Social Semiotic. Edward Arnold, London, 1978.

---, "Modality and modulation in English." In (Kress, 1976, Ch. 13), 1970.

Halliday, M.A.K. & Martin,J.R. (eds.) Readings in Systemic Linguistics. Batsford Academic, London, 1981.

Hasling,D., Clancey,W., Rennels,G., "Strategic explanation for a diagnostic consultation system." In Coombs,M.(ed.) Developments in Expert Systems. Academic Press, London, 1984, pp. 117-133.

Hayes-Roth,F., Waterman,D., Lenat,D., (eds.) Building Expert Systems. Addison-Wesley, London, 1983.

deJoia,A. & Stenton,A. Terms in Systemic Linguistics. Batsford Academic, London, 1980.

Kress,G. (ed.) Halliday: System and Function in Language. Oxford, London, 1976.

Mann,W./Halliday,M.A.K. "Systemic Grammar of English, S.G. Clause Systems". From the PENMAN system, Information Sciences Institute, USC.

Mann,W. & Matthiessen,C. "Nigel: A Systemic Grammar for Text Generation". RR-83-105, Information Sciences Institute, USC. 1983.

Monaghan,J. The Neo-Firthian Tradition and its Contribution to General Linguistics. Max Niemeyer Verlag, Tubingen, 1979.

Stefik,M., Aikins,J., Balzer,R., Benoit,J., Birnbaum,L., Hayes-Roth,F., Sacerdoti,E., "The architecture of expert systems." In (Hayes-Roth et al., 1983), 1983.

Winograd,T. Language as a Cognitive Process. Addison-Wesley, London, 1983.