ON COMPUTATIONAL SENTENCE GENERATION FROM LOGICAL FORM

Juen-tin Wang
Institut für Angewandte Informatik, TU Berlin

Summary.
This paper describes some computational and linguistical mechanisms in our program written in SIMULA to generate natural language sentences from their underlying logical structures in an extended predicate logic. After the presentation of the augumented logical formalism to deal with illocutionary acts,we explain then the basic devices used in the generation process:semiotic interpretation,orders of quantifications or derivational constraints,the referential property of variables and the Leibniz-Frege idea.Examples from system output will be given.

1.Introduction

Logical form is one of the most used notions in philosophy,logic and linguistics. It goes back at least to Aristoteles in his linguistical and logical analysis of natural language sentences. This direct reference to the immediate sentence form which has been characteristic for the logic of syllogism remains unchanged throughout the whole period of scholastic logic until the development of the formal predicate logic. Since then,this logical formalism, with or without variation and modification, has been widely used in the linguistic philosophy to analyse and study the natural language. And it is then the resulted representations in logical formalism which will be taken as the logical form of the analyzed natural language sentences. This changed notion of logical form can be found everywhere in the tractatus of Carnap,Quine,Geach,Hintikka and many others. And this notion of logical form will be now used universally. In recent times, a lot of logically minded linguists like Lakoff, Harman,Keenan and Karttunen have even attempted to put logical form into the relationship with the notion of deep structure in connection with Chomsky's theory of generative grammar. They hold the view that the semantical representation of natural language sentences can be obtained from the formal logical structures and that these semantical representations can be adapted as a basis for sysntactical generation of natural language sentences. However,this school of generative grammar has not given any constructive demonstration of their assertions.

In this paper we do not concern with the question whether this theory,the so-called generative semantics ,will yield a true grammar theory or a genuine theory of the semantics of natural language. We are rather motivated by real needs. We have already at our disposal a question-answering data base system which uses essentially the language of predicate logic as the formal query language. We need to know how to express these logical forms in natural language sentences. And Since we have to do with a question-answering system, we need not only to treat logical forms underlying indicative sentences but,more important, the logical structures which have been used by the system as the representations for interrogative sentences.In the following we present at first the extended logical formalism. We describe then the conceptions and principles being used in implementation. The program is written in SIMULA.

2.Logical formalism as semantical representation of natural language sentences

The logical formalism which we have used to represent the sentence structure of a natural language is in its essence a many-sorted language of predicate logic. In the conception of representation we have adopted some ideas from the speech act theory of Austin and Searle.According to this theory, the utterance of any sentences in a natural language is characteristically performing at least three distinct kinds of acts:(1) the uttering of words,(2) referring and predicating, (3) stating, questioning, commanding, promising,etc. The notion of referring and predicating should be thus detached from the notions of such speech acts as asserting,questioning, commanding,etc. , since the same reference and predication can occur in the performance of different complete speech acts. In taking account of this distinction between proposition and illocutionary act we make one addition to the usual logical formalism. We let the pro-

positional part be represented by the usual logical expression. In addition, we have an auxiliary component to represent the different illocutionary acts. This additional component will be connected with the left end of the logical expression by a convention sign " = ", which,by the way, should not be read as "equal". A detailed description of this extended logical formalism is given in Habel,Schmidt and Schweppe (1977). Some examples can be given as follows:

```
Assertions:
  /true/=.all.x2(.ex.x3(city(x3).eq.
  `Tokyo´.and. takeplacecity(x2,x3)))
                                    (1)
Requests(Wh-questions):
  conference(x2)=.ex.x3(city(x3).eq.
  `Tokyo´ .and. takeplacecity(x2,x3))
                                    (2)
yes-no questions:
  =.all.x2(.ex.x3(city(x3).eq.`Tokyo´
   .and. takeplacecity(x2,x3)))
                                    (3)
```

The illocutionary indicators like "conference(x2)", which is itself a name function, can be compared with the designator of Woods(1968) in his query language formalism. In general, several such illocutionary indicators can be allowed at the same time; they could then lead to the representation of multiple questions as discussed by Hintikka. Here,however, we leave the question open, whether this proposed logical formalism as a representation symbolism is complete and adequate for natural language. For example, we do not consider whether WHY- and HOW-question can also be treated in the same framework.

It is obvous that this proposal for the semantical representation of natural language sentences does not follow Chomsky´s theory, according to which interrogative sentences should be derived from non-interrogative ones by the application of optional transformations. This approach has rather some affinity with the suggestion of Ajdukiewicz(1928) who has described the logical structure of a question as consisting of sentential matrix(a sentence with one or more of its components replaced by variables) preceded by an interrogative operator `for what x´(or `for what x,y ,z,...´,if the matrix has more than one free variable). In such cases, we can take illocutionary indicators as interrogative operators in the sense of Ajdukiewicz.

The proposed way of giving semantical representations both to indicative and question sentences seems to have some advantages. Above all, it enable us to deal with question sentences directly without using the somehow artificial method to paraphrase them as indicative sentences or spistemic statements, as suggested by Hintikka. Any way, the suggested kind of semantical representation of question sentences receives a quite natural set-theoretical interpretation. For example, the form (2) used for request corresponds to the meaning of the set expression:

$$\left\{ x2 \mid .ex.x3(city(x3).eq.`Tokyo´.and. takeplacecity(x2,x3)) \right\} \qquad (4)$$

In such cases, the interrogative operators function as quantifiers; they bind free variables and thus transform conditions exhibited in sentential matrix into complete closed forms.

3. Examples

In order to let the reader have a rough impression of what the system can accomplish at the present stage, we give below at first some output examples, before we step into the scattered description of the conceptions and principles to be used. The examples taken from output consists of pairs of a given logical form and its corresponding natural language sentence generated.

```
/true/=.all.x2(.ex.x3(city(x3).eq.
  `Tokyo´.and. takeplacecity(x2,x3)))
EVERY MEETING WILL BE HELD IN THE
CITY`Tokyo´

/true/=.all.x1(.ex.x2((.ex.x3(make-
journeycity(x1,x3).and.city(x3).eq.
`tokyo´)).imp.(takepart(x1,x2))))
EVERYBODY WHO MAKES A JOURNEY TO THE
CITY`TOKYO´ TAKES PART AT SOME MEETING

person(x1)=.ex.x2((.ex.x4(takeplace-
country(x2,x4).and.country(x4).eq.
`japan´)).and.(takepart(x1,x2)))
WHO TAKES PART AT THE MEETING WHICH
TAKES PLACE IN THE COUNTRY `JAPAN´

conference(x2)=.ex.x3(city(x3).eq.
`tokyo´.and. takeplacecity(x2,x3))
WHICH MEETINGS WILL BE HELD IN THE
CITY `TOKYO´

person(x1)=.ex.x2(conference(x2).eq.
`colling-8o´.and.(.ex.x3(takeplace-
city(x2,x3).and.city(x3).eq.`tokyo´))
.and.givelectureconf(x1,x2))
WHO GIVES A LECTURE AT THE CONFERENCE
`COLLING-8o´ WHICH TAKES PLACE IN THE
CITY `TOKYO´

country(x4)=.ex.x2(takeplacecountry
(x2,x4).and.conference(x2).eq.`colling
-8o´)
IN WHICH COUNTRIES WILL THE MEETING
```

`COLLING-8o´ BE HELD

```
person(x1)=.ex.x3(city(x3).eq.`tokyo´
.and.(.ex.x2(takeplacecity(x2,x3).and.
conference(x2).eq.`colling-8o´)).and.
traveltocity(x1,x3))
```

WHO TRAVELS TO THE CITY `TOKYO´ IN WHICH THE MEETING `COLLING-8o´ TAKES PLACE

```
person(x1)=.ex.x2((.ex.x4(takeplaceco-
untry(x2,x4).and.country(x4).eq.`japan´
)).and.(takepart(x1,x2)))
```

WHO TAKES PART AT THE MEETING WHICH TAKES PLACE IN THE COUNTRY `JAPAN´

4. Semiotic interpretation as sentence generation basis

Let us proceed to consider the devices for sentence generation from the underlying logical structure. Essentially the generation process will be based on the semiotic interpretation,called by Scholz and Hasenjaeger, of the predicates and functions used in the logical structure. Some of them are listed as follows:

Predicates:

takepart(x1,x2) $=_{def}$ person x1 takes part at meeting x2

takeplacecity(x1,x2) $=_{def}$ meeting x1 will be held in city x2

takeplacecountry(x1,x2) $=_{def}$ meeting x1 takes place in country x2

makejourneycity(x1,x2) $=_{def}$ person x1 makes a journey to city x2

Functions:

city(x).eq.y $=_{def}$ the name of city x is y

conference(x).eq.y $=_{def}$ the name of meeting x is y

person(x).eq.y $=_{def}$ the name of person x is y

The semiotic interpretation strings are the building basis for surface sentences. In this respect the semiotic interpretation of predicate may be comparable with the underlying string in the generation tree or phrase-marker which is assumed both in the theory of Chomsky and in the theory of Montague as well. If we look at its actual form more closely, the strings given as semiotic interpretations differ in one essential point from the underlying strings adopted in the school of generative grammar. The underlying string in the deep structure for grammatical transformation contains no variable as used in the logic. On the ground of this essential difference we can make no direct comparison between our approach and that of generative semantics.

At the disposal of semiotic interpretations of predicates and functions, we could already in principle implement a program to generate somehow quasi natural language sentences from the given logical structures. All what we need to do is to follow the type of reading the logical formula which we have been taught at the class room. We have been taught, for example, to read the following logical structure

```
/true/=.all.x2(.ex.x3(city(x3).eq.
`tokyo´.and.takeplacecity(x2,x3)))
```

as:

for every meeting x2 it holds:
there is a city x3,for which it hold:
the name of city x3 is `tokyo´
and
meeting x2 will be held in city x3

This might be considered as a quasi natural language sentence formulation. It has above all the advantage of being universal to the extent that it can be applied to every kind of logical structures. And actually a program has worked in this style (Habel,Schmidt,Schweppe 1977). However,this kind of formulation is not the usual surface sentence and it is also not so intelligible as it could. We need therefore to find out an alternative which might give us a simple and natural formulation . For eample, the logical form given above has the meaning which can be expressed simply as:

"Every meeting will be held in the city `Tokyo´ "

It contains no formal logical quantifiers and no free or bounded variables. We describe below some main methods and principles which we have used to achieve the generation of such surface sentences computationally.

5. Quantification order and derivational constraint

The problem of quantifiers constitutes one of major obstacles in the computational sentence generation from logical structures.As is well known, the order of different quantifiers has an influence on the meaning of the expression whether it is in the case of natural language or it is in the case of predi-

cate logic. Thus, Peirce has already pointed out that the sentences

"some woman is adored by whatever spaniard may exist"
and
"whatever spaniard my exist adores some woman"

have quite different meanings. Hintikka and Lakoff have made the same observation in their analysis of natural language (but it seems that Chomsky has overlooked this fact in his formulation of Passive-transformation).This phenomenon that the order in which universal and particular quantifier occur is material for the meaning is even more obvious in the language of predicate logic.

Let us consider as example the predicate

personvisitcity(x,y)

with the assigned semiotic interpretation:

person x visits city y .

The two logical expressions

.all.x1(.ex.x2(personvisitcity(x1,x2)))
.ex.x2(.all.x1(personvisitcity(x1,x2)))

which differs from each other just in the order of quantification means quite differently. In the process of sentence generation from logical structure we can thus not simply take the semiotic interpretation string and substitute for its variables the corresponding types of quantifiers. In other words, the operation of "quantifier-lowering", as Lakoff has called it, can not be applied in all cases without pertinent differentiation. In our example, it can be applied in the first case and yields the correct sentence:

"every person visits some city " .

However,its direct application would lead rather to incorrect sentence in respect to the secand logical form. It has rather the meaning

"some city will be visited by every person " .

The regularity for the possibility of substitution can be perceived if we look at the semiotic interpretation string and consider the patterns of the following logical forms together:

I.
.all.x1(.ex.x2(personvisitcity(x1,x2)))
.ex.x1(.all.x2(personvisitcity(x1,x2)))

II.
.all.x2(.ex.x1(personvisitcity(x1,x2)))
.ex.x2(.all.x1(personvisitcity(x1,x2)))

It is then obvious that only in cases, while the order of logical quantifiers is in the same sequence in which the corresponding variables occur in the given semiotic interpretation, the operation of quantifier-lowering can be directly carried out. And it yields correct sentences. In other cases such as in (II),it is without measures not possible. This kind of regularity has been also observed by Lakoff in his discussion of the notion of derivational ;it occurs in the transformational derivation of surface sentences from the underlying deep structures. Without going into the details of his final modifications,the derivational constraint means roughly like this: if one quantifier commands another in underlying structure, then that quantifier must be leftmost in surface structure. He uses the derivational constraint as a means to rule out certain kind of transformational generation of incorrect surface sentences. Our aim is ,however, not to block out but to obtain correct and meaningful surface sentences from meaningful logical structures. We thus try to find out means so that the condition of derivational constraint can always,or at least to a large part, be fulfilled. For this purpose we introduce the notion of the associated forms of the semiotic interpretation of the given predicate. We add for example to the original semiotic interpretation

"person x visits city y" (5)

its associated form like

"city y will be visited by person x" (6).

It will be simply stored. In dependence on the orders of quantifiers the corresponding semiotic interpretation string will be selected. By this additional means, correct sentences could then be computationally generated from the logical patters mentioned in (II).

The same problem occurs with the treatment of logical structures underlying Wh-questions (which, who, etc.,). In our conception and in accordance also with the theory of Hintikka,the interrogative operators has the quantification nature. They subject thus to the same derivational constraints. We use thus the associated semiotic interpretation strings in the required cases. By this means, we can generate computationally from the logical structures

person(x1)=.all.x2(personvisitcity(x1,x2))
city(x2)=.all.x1(personvisitcity(x1,x2

))
the following interrogative sentences respectively:

"Who visits every city" ,
"Which cities will be visited by every person"

It is of interest to note that with this device the topic of interrogative sentences has been treated and solved for the simple cases at the same time. In general, the device of associated forms of the semiotic interpretation,which from the linguistical viewpoint relate to each other transformationally, will be extensively used. Among others, it will be applied in the treatment of the relative sentences. In other words,associated form like

"who makes a journey to city y "

will be stored together with the given interpreted predicate; and this associated form will be used eventually for relative sentence formation.We return to this problem below.

6. Referential property of variable, relative sentence generation and and property of connectivity

In computational sentence generation from the underlying logical structure we make an extensive use of the reference nature of the variables. Variables have been called by Quine as pronouns of logic and mathematics. The referential character will be used by us as a kind of red thread in building up the composed sentences. This feature shows clearly in generating sentences with relative clauses. Let us consider as example the logical structure

```
person(x1)=.ex.x2((.ex.x4(takeplace-
country(x2,x4).and.country(x4).eq.
`japan´)).and.(takepart(x1,x2)))
```

The variable x1 in the interrogative operator, namely person(x1), indicates the topic of the question concerned. This topic is in general specified by the composition of predicates and functions in a certain way which is expressed by the logical matrix.The generation of the corresponding interrogative sentence means to express verbally this composition of predicates and functions after the given prescription in matrix. In making use of the referential property of variavles, it is seen that the topic will be characterized at first by the predicate

takepart(x1,x2)

On this ground its associated form of semiotic interpretation, namely

"who takes part at meeting x2"

will be used as the main building component of the question sentence to be generated. By means of the variable, we can find that this predicate

takepart(x1,x2)

is connected directly with the predicate

takeplacecountry(x2,x4) .

In other words, the variable x2 contained in the predicate

takepart(x1,x2)

is in its turn specified by the predicate takeplacecountry(x2,x4). We use thus in consideration of its modification character the corresponding associated form of semiotic interpretation, namely

"which takes place in country x4"

to build up the relative clause. In the same way, we find that the variable x4 contained in the predicate

takeplacecountry(x2,x4)

is referred by the name function

country(x4),

whose function value indicates the name Japan. This constant will be thus inserted at the place x4. The termination of these connecting and inserting processes lead then to the generation of the sentence

"Who takes part at the meeting which takes place in the country `Japan´".

In connection with the referential feature of variables it is of interest to note that all the logical structures which we have used in our question-answering system shows a remarkable property which we have called the property of connectivity. A logical structure is called to have the property of connectivity, if in the case where it contains more than one predicate or function each of its predicates and functions shares some argument with others,i.e. has common variables with other functions or predicates.

It is on the ground of the property of this connectivity that we can even let the program processing under certain circumstances be driven by variables, such as explained just above. On the contrary, let us consider the following logical structure:

```
/true/=.ex.x1(.ex.x2(.ex.x3(city(x3)
.eq.`tokyo´.or.takepart(x1,x2))))
```

Since the function city(x3) and the predicate takepart(x1,x2) do not share any common argument, this logical form

does not have the defined property of connectivity. Its corresponding surface sentence can therefore not be computed by the process driven by variables. Instead, a different procedure must be applied. At present stage, we let, however, such types of sentences out of our consideration.

The usefulness of variables is not exhausted in relative sentence generation. In general, we intend to use it to differentiate the varied patterns of the logical forms concerned. And as a result of this differentiation, sentences of varied patterns will be generated. Let us consider the following simple logical form:

```
person(x1)=.ex.x2(takepart(x1,x2).and.
givelectureconf(x1,x2))
```

For such pattern, no attempt to generate relative sentence will be made. Instead, it tries to express the surface sentence as follows:

"Who takes part at some meeting and gives a lecture at this meeting " .

Our program is thus in trying to discern as much of logical patterns as possible. It works after them.

7. Categorical and hypothetical sentences, idea of Leibniz and Frege

In our computational sentence generation we have made use of an old idea, which goes back at least to an observation made by Leibniz in his famous nouveau essais sur lèntendement humain. In the classical logic, one is customed namely to divide the judgements or assertional indicative sentences into three major types:

categorical,hypothetical and disjunctive .

Leibniz has remarked that in some cases an actual hypothetical judgement can be expressed in a categorial form. This regularity is discussed also by Frege on the relation between auxiliary sentences (Beisätze) and conditional sentences (Bedingungssätze)in his essay über Sinn und Bedeutung. According to Frege the cinditional or hypothetical sentence

"Wenn eine Zahl kleiner als 1 und größer als o ist, so ist auch ihr Quadrat kleiner als 1 und größer als o "

can be expressed in a categorial form:

"Das Quadrat einer Zahl, die kleiner als 1 and größer als o ist, ist kleiner als 1 und größer als o " .

In our system design, we have adopted this old conception. From the underlying logical implication structure its surface sentence will not be generated in hypothetical, but rather in categorial form. This approach has its practical and stylistic advantages. It can be seen in consideration of the following logical form:

```
/true/=.all.x1(.ex.x2((.ex.x3(
makejourneycity(x1,x3).and.city(x3)
.eq.`tokyo´)).imp.(takepart(x1,x2)))).
```

In following this line of thought,the corresponding surface sentence will be generated by the system as follows:

"Everybody who makes a journey to the city `Tokyo´ takes part at some meeting "

It is natural and simple. For its generation we need no more additional methods than the ones which have been at our disposal:the quantifier-lowering and formation of relative sentence. The only thing which we must take care of is to choose the semiotic interpretation string of the conclusion rather than that of antecedent as the main building component. Otherwise, the meaning would be distorted.

The usefulness of this conception of Leibniz and Frege consists for our purpose, above all, in the fact that it can be even extended to the treatment of logical structures for interrogative sentences. Without using this idea,the surface sentences to be computationally generated would have a cumbersome look. This feature may appear clearly,if we try to deal with the following simple logical structure:

```
conference(x2)=.all.x1((.ex.x3(
makejourneycity(x1,x3).and.city(x3)
.eq.`tokyo´)).imp.(takepart(x1,x2)))
```

It is a logical form underlying an interrogative sentence; it contains the logical form mentioned just above almost as component. In combination of this Leibniz-Frege idea with the other principles like referential property of variables, topic handling and formation of relative sentence which we have described above the system yields then without other detour the interrogative sentence:

"Which meetings will be visited by everyone who makes a journey to the city `Tokyo´ "

8. General remark and discussion

We have above described some main conceptions and principles upon which we have built up the program. The system works essentially after logical patterns, after certain features of logical structures such as connectivity, the occurence of implication sign and so on. It is thus properties-oriented and not syntax-driven. It is needless to say that our program can not deal with all kinds of logical structures. This is also not our original aim, besides the fact that,as Chomsky makes remark about the nature of deep structures, not all logical structure can underly or have a meaningful surface sentence. From the right beginning we have confined ourself to just a specified set of logical structures used as a formal query language. It is remarkable that for such a set of logical forms certain regularities and patterns can be generally established and be used to generate meaningful surface sentences computationally. The progress will depend to a large extent on the careful observation of logical patterns and insightful linguistic analyses.

9. References

Frege,G.:
Über Sinn und Bedeutung, Ztschr.f. Philos.u.philos.Kritik,NF 1oo,1892.

Habel,Ch.,Schmidt,A.,Schweppe,H.:
On automatic paraphrasing of natural language expressions, Semantic Network project report 3/77, 1977, TU Berlin.

Hintikka,J.:
The semantics of questions and the questions of semantics, Amsterdam, North-Holland, 1976.

Lakoff,G.
On generative semantics, in: D.D. Steinberg et al(eds.): Semantics, Cambridge,uni.press,1971

Searle,J.R.:
Speech acts, Cambridge,1969.