FUNCTIONAL UNIFICATION GRAMMAR: A FORMALISM FOR MACHINE TRANSLATION

Martin Kay

Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto
California 94304

and

CSLI, Stanford

Abstract

Functional Unification Grammar provides an opportunity to encompass within one formalism and computational system the parts of machine translation systems that have usually been treated separately, natably analysis, transfer, and synthesis. Many of the advantages of this formalism come from the fact that it is monotonic allowing data structures to grow differently as different nondeterministic alternatives in a computation are pursued, but never to be modified in any way. A striking feature of this system is that it is fundamental reversible, allowing *a* to translate as *b* only if *b* could translate as *a*.

I Overview

A. Machine Translation

A classical translating machine stands with one foot on the input text and one on the output. The input text is analyzed by the components of the machine that make up the left leg, each one feeding information into the one above it. Information is passed from component to component down the right leg to construct the output text. The components of each leg correspond to the chapters of an introductory textbook on linguistics with phonology or graphology at the bottom, then syntax, semantics, and so on. The legs join where languages are no longer differentiated and linguistics shades off into psychology and philosophy. The higher levels are also the ones whose theoretical underpinnings are less well known and system designers therefore often tie the legs together somewhere lower down, constructing a more or less *ad hoc* bridge, pivot, or transfer component.

We connot be sure that the classical design is the right design, or the best design, for a translating machine. But it does have several strong points. Since the structure of the components is grounded in linguistic theory, it is possible to divide each of these components into two parts: a formal description of the relevant facts about the language, and an interpreter of the formalism. The formal description is data whereas the interpreter is program. The formal description should ideally serve the needs of synthesis and analysis indifferently. On the other hand we would expect different interpreters to be required in the two legs of the machine. We expect to be able to use identical interpreters in corresponding places in all machines of similar design because the information they embody comes from general lingusitic theory and not from particular languages. The scheme therefore has the advantage of modularity. The linguistic descriptions are independent of the leg of the machine they are used in and the programs are independent of the languages to which they are applied.

For all the advantgages of the classical design, it is not hard to imagine improvements. In the best all possible worlds, there would only be one formalism in which all the facts about a language—morphological, syntactic, semantic, or whatever—could be stated. A formalism powerful enough to accommodate the various different kinds of linguistic phenomena with equal facility might be unappealing to theoretical linguists because powerful formal systems do not make powerful claims. But the engineering advantages are clear to see. A single formalism would straightforwardly reduce the number of interpreters to two, one for analysis and one for synthesis. Furthermore, the explanatory value of a theory clearly rests on a great deal more than the restriciveness of its formal base. In particular, the possiblity of encompassing what had hitherto been thought to require altogether different kinds of treatment within a single framework could be theoretically interesting.

Another clear improvement on the classical design would result from merging the two interpreters associated with a formalism. The most obvious advantage to be hoped for with this move would be that the overall structure of the translating machine would be greatly simplified, though this would not *necessarily* happen. It is also reasonable to hope that the machine would be more robust, easier to modify and maintain, and altogether more perspicuous. This is because a device to which analysis and synthesis look essentially the same is one that is fundamentally less time dependent, with fewer internal variables and states; it is apt to work by monitoring constraints laid down in the formal description and ensuring that they are maintained, rather than carrying out long and complex sequences of steps in a carefully prescribed order.

These advantages are available in large measure through a class of formal devices that are slowly gaining acceptance in linguistics and which are based on the relations contracted by formal objects rather than by transformations of one formal object into another. These systems are all procedurally *monotonic* in the sense that, while new information may be added to existing data structures, possibly different information on different branches of a nondeterministic process, nothing is ever deleted or changed. As a result, the particular order in which elementary events take place is of little importance. Lexical Functional Grammar and Generalized Phrase-Structure grammar share these relational and monotonic properties. They are also characteristics of Functional Unificational Grammar (FUG) which I believe also has additional properties that suit it particularly well to the needs of experimental machine-translation systems.

The term *experimental* must be taken quite seriously here though, if my view of machine translation were more generally held, it would be redundant. I believe that all machine translation of natural languages is experimental and that he who claims otherwise does his more serious colleagues a serious disservice. I should not wish any thing that I say in this paper as a claim to have solved any of the miriad problems that stand between us and working machine translation systems worthy of the name. The contribution that FUG might make is, I believe, a great deal more

modest, namely to reformalize more simply and perspicuously what has been done before and which has come to be regarded, as I said at the outset "classical".

B. Functional Unification Grammar

FUG traffics in descriptions and there is essentially only one kind of description, whether for lexical items, phrases, sentences, or entire languages. Descriptions do not distinguish among levels in the linguistic hierarchy. This is not to say that the distinctions among the levels are unreal or that a linguist working with the formalism whould not respect them. It means only that the notation and its interpretation are always uniform. Either a pair of descriptions is incompatible or they are combinable into a single description.

Within FUG, every object has infinitely many descriptions, though a given grammar partitions the descriptions of the words and phrases in its language into a finite number of equivalence classes, one for each interpretation that the grammar assigns to it. The members of an equivalence class differ along dimensions that are grammatically irrelevant—when they were uttered, whether they ammused Queen Victoria, or whether they contain a prime number of words. Each equivalence class constitutes a lattice with just one member that contains none of these grammatically irrelevant properties, and this *canonical* member is the only one a linguist would normally concern himself with. However, a grammatical irrelevancy that acquires relevance in the present context is the description of possible translations of a word or phrase, or of one of its interpretations, in one or more other languages.

A description is an expression over an essentially arbitrary basic vocabulary. The relations among sets of descriptions therefore remain unchanged under one-for-one mappings of their basic vocabularies. It is therefore possible to arrange that different grammars share no terms except for possible quotations from the languages described. Canonical descriptions of a pair of sentences in different languages according to grammars that shared no terms could always be unified into a single description which would, of course, not be canonical. Since all pairs are unifiable, the relation that they establish between sentences is entriely arbitrary. However, a third grammar can be written that unifies with these combined descriptions only if the sentences they describe in the two langauages stand in a certain relation to one another. The relation we are interested in is, of course, the translation relation which, for the purposes of the kind of experimantal system I have in mind I take to be definable even for isolated sentences. Such a *transfer* grammar can readily capture all the components of the translation relation that have in fact been built into translation systems: correspondences between words and continuous or discontinuous phrases, use of *selectional features* or local contexts, case frames, reordering rules, lexical functions, compositional semantics, and so on.

II The Formalism

A. Functional Descriptions

In FUG, linguistic objects are represented by *functional descriptions* (FDs). The basic constituent of a functional description is a *feature* consisting of an *attribute* and an associated *value*. We write features in the form $a = v$, where a is the attribute and v, the value. Attributes are arbitrary words with no significant internal structure. Values can be of various types, the simplest of which is an *atomic value*, also an arbitrary word. So *Cat* = *S* is a feature of the most elementary type. It appears in the descriptions of sentences, and which declares that their *Category* is *S*. The only kinds of non-atomic values that will concern us here are *constituent sets*, *patterns* and FDs themselves.

A FD is a Boolean expression over features. We distinguish conjuncts from disjuncts by the kinds of brackets used to enclose their members; the conjuncts and disjuncts of $a = p$, $b = q$, and $c = r$ are written

$$\begin{bmatrix} a = p \\ b = q \\ c = q \end{bmatrix} \quad \text{and} \quad \begin{Bmatrix} a = p \\ b = q \\ c = r \end{Bmatrix}$$

respectively. The vertical arrangement of these expressions has proved convenient and it is of minor importance in that braces of the ordinary variety are used for a different purpose in FUG, namely to enclose the members of consituent sets. The following FD describes all sentences whose subject is a singular noun phrase in the nominative or accusative cases

$$(1) \qquad \begin{bmatrix} \text{Cat} = \text{S} \\ \text{Subj} = \begin{bmatrix} \text{Cat} = \text{NP} \\ \text{Num} = \text{Sing} \\ \begin{Bmatrix} \text{Case} = \text{Nom} \\ \text{Case} = \text{Acc} \end{Bmatrix} \end{bmatrix} \end{bmatrix}$$

It is a crucial property of FDs that no attribute should figure more than once in any conjunct, though a given attribute may appear in feature lists that are themselves the values of different attributes. This being the case, it is always possible to identify a given conjunct or disjunct in a FD by giving a sequence of attributes $(a_1 \cdots a_k)$. a_1 is a attribute in the FD whose value, v_1, is another FD. The attribute a_2 is an attribute in v_1 whose value if an FD, and so on. Sequences of attributes of this kind are referred to as *paths*. If the FD contains disjuncts, then the value identified by the path will naturally also be a disjunct.

We sometimes write a path as the value of an attribute to indicate that that value of that attribute is not only eaqual to the value identified by the path but that these values are one and the same, inshort, that they are *unified* in a sense soon to be explained. Roughly, if more information were acquired about one of the values so that more features were added to it, the same additions would be reflected in the other value. This would not automatically happen because a pair of values happened to be the same. So, for example, if the topic of the sentence were also its object, we might write

$$\begin{bmatrix} \text{Object} = v \\ \text{Topic} = (\textit{Object}) \end{bmatrix}$$

where v is some FD.

Constituent sets are sets of paths identifying within a given FD the descriptions of its constituents in the sense of phrase-structure grammar. No constituent set is specified in example (1) above and the question of whether the subject is a constituent is therefore left open.

Example (2), though still artificially simple, is more realistic. It is a syntactic description of the sentence *John knows Mary*. Perhaps the most striking property of this description is that descriptions of constituents are embedded one inside another, even though the constituents themselves are not so embedded. The value of the *Head* attribute describes a constituent of the sentence, a fact which is declared in the value of the *CSet* attribute. We also see that the sentence has a second attribute whose decription is to be found as the value of the Subject of the Head of the Head of the sentence. The reason for this arrangement will become clear shortly.

In example (2), every conjunct in which the *CSet* attribute has a value other than *NONE* also has a substantive value for the attribute *Pat*. The value of this attribute is a regular expression over paths which restricts the order in which the constituents must appear. By convention, if no pattern is given for a description which nevertheless does have constituents, they may occur in any order. We shall have more to say about patterns in due course.

B. Unification

Essentially the only operation used in processing FUG is that of *Unification*, the paradigm example of a monotonic operation. Given a pair of descriptions, the unification process first determines whether they are compatible in the sense of allowing the possibility of there being some object that is in the extension of both of them. This possibility would be excluded if there were a path in one of the two descriptions that lead to an atomic value while the same path in the other one lead to some other value. This would occur if, for example, one described a sentence with a singular subject and the other a sentence with a plural subject, or if one described a sentence and the other a noun phrase. There can also be incompatibilities in respect of other kinds of value. Thus, if one has a pattern requiring the subject to precede the main verb whereas the other specifies the other order, the two descriptions will be incompatible. Constituent sets are incompatible if they are not the same.

We have briefly considered how three different types of description behave under unification. Implicit in what we have said is that descriptions of different types do not unify with one another. Grammars, which are the descriptions of the infinite sets of sentences that make up a language constitute a type of description that is structurally identical an ordinary FD but is distinguished on the grounds that it behaves slightly differently under unification. In particular, it is possible to unify a grammar with another grammar to produce a new grammar, but it is also possible to unify a grammar with a FD, in which case the result is a new FD. The rules for unifying grammars with grammars are the same as those for unifying FDs with FDs. The rules for unifying grammars with FDs, however, are slightly different and in the difference lies the ability of FUG to describe structures recursively and hence to provide for sentences of unbounded size. The rule for unifying grammars with FDs requires the grammars to be unified—following the rules for FD unification—with *each individual* constituent of the FD.

(3)
$$\left\{\begin{bmatrix}\text{Head} = [\text{Head} = [\text{Cat} = \text{V}]] \\ \text{CSet} = \{\langle\text{Head Head Subj}\rangle\langle\text{Head}\rangle\} \\ \text{Pat} = (\langle\text{Head Head Subj}\rangle\langle\text{Head}\rangle)\end{bmatrix}\right.$$
$$\left.\begin{bmatrix}\text{Head} = \begin{bmatrix}\text{Cat} = \text{V} \\ \{\text{Obj} = \text{NONE} \\ \text{Obj} = [\text{Cat} = \text{NP}]\} \\ \text{CSet} = \text{NONE}\end{bmatrix}\end{bmatrix}\right\}$$
$$\left\{\begin{bmatrix}\text{Head} = \begin{bmatrix}\text{Cat} = \text{N} \\ \text{CSet} = \text{NONE}\end{bmatrix}\end{bmatrix}\right\}$$

By way of illustration, consider the grammar in (3). Like most grammars, it is a disjunction of clauses, one for each (nonterminal) category or constituent type in the language. The first of the three clauses in the principle disjunction describes sentences as having a head whose head is of category *V*. This characterization is in line with so called $\overline{X}$-theory, according to which a sentence belongs to the category $\overline{\overline{\text{V}}}$. In general, a phrase of category $\overline{X}$, for whatever X, has a *head* constituent of category X, that is, a category with the same name but one less bar. $\overline{X}$ is built into the very fabric of the version of FUG illutrated here where, for example, a setence is by definition a phrase whose head's head is a verb. The head of a sentence is a $\overline{\text{V}}$, that is, a phrase whose head is of category V and which has no head of its own. A phrase with this description cannot unify with the first clause in the grammar because its head has the feature [Head = NONE].

Of sentences, the grammar says that they have two constituents. It is no surprise that the second of these is its head. The first would usually be called its subject but is here characterized as the subject of its verb. This does not imply that there must be lexical entries not only for all the verbs in the language but that there must be such an entry for each of the subjects that the verb might have. What it does mean is that the subject must be unifiable with any description the verb gives of its subject and thus provides automatically both for any selectional restrictions that a verb might place on its subject but also for agreement in person and number between subject and verb. Objects are handled in an analogous manner. Thus, the lexical entries for the French verb forms *connaît* and *sait* might be as follows:

$$\begin{bmatrix}\text{Cat} = \text{V} \\ \text{Lex} = \text{connaitre} \\ \text{Tense} = \text{Pres} \\ \text{Subj} = \begin{bmatrix}\text{Pers} = 3 \\ \text{Num} = \text{Sing} \\ \text{Anim} = +\end{bmatrix} \\ \text{Obj} = [\text{Cat} = \text{NP}]\end{bmatrix} \qquad \begin{bmatrix}\text{Cat} = \text{V} \\ \text{Lex} = \text{savoir} \\ \text{Tense} = \text{Pres} \\ \text{Subj} = \begin{bmatrix}\text{Pers} = 3 \\ \text{Num} = \text{Sing} \\ \text{Anim} = +\end{bmatrix} \\ \text{Obj} = [\text{Cat} = \text{S}]\end{bmatrix}$$

Each requires its subject to be third person, singular and animate. Taking a rather simplistic view of the difference between these verbs for the sake of the example, this lexicon states that *connaît* takes noun phrases as objects, whereas *sait* takes sentences.

III Translation

A. Syntax

Consider now the French sentence *Jean connaît Marie* which is presumably a reasonable rendering of the English sentence *John knows Mary*, a possible fumctional description of which we was given in (2). I take it that the French sentence has an essentially isomorphic structure. In fact, following the plan laid out at the beginning of the paper, let us assume that the functional description of the French sentence is that given in (2) with obvious replacements for the values of the *Lex* attribute and with attribute names x_i in the English grammar systematically replaced by *F-*x_i in the French. Thus we have *F-Cat*, *F-Head*, etc. Suppose now, that, using the English grammar and a suitable parsing algorithm, the structure given in (2) is derived from the English sentence, and that this description is then unified with the following *transfer* grammar:

$$\begin{bmatrix}\text{Cat} = \langle\text{F-Cat}\rangle \\ \left\{\begin{bmatrix}\text{Lex} = \text{John} \\ \text{F-Lex} = \text{Jean}\end{bmatrix}\right. \\ \left.\begin{bmatrix}\text{Lex} = \text{Mary} \\ \text{F-Lex} = \text{Marie}\end{bmatrix}\right. \\ \left\{\begin{bmatrix}\text{Lex} = \text{know} \\ \{\text{F-Lex} = \text{connaitre} \\ \text{F-Lex} = \text{savoir}\}\end{bmatrix}\right\}\end{bmatrix}$$

The first clause of the principal conjunct states a very strong requirement, namely that the description of a phrase in one of the two languages should be a description of a phrase of the same category in the other language. The disjunct that follows is essentially a bilingual lexicon that requires the description of a lexical item in one language to be a description of that word's counterpart in the other language. It allows the English verb *know* to be set in correspondence with either *connaître* or *savoir* and gives no means by which to distinguish them. In the simple example we are developing, the choice will be determined on the basis of criteria expressed only in the French grammar, namely whether the object is a noun phrase or a sentence.

This is about as trivial a transfer grammar as one could readily imagine writing. It profits to the minimal possible extent from the power of FUG. Nevertheless, it should already do better than word-for-word translation because the transfer grammar says nothing at all about the order of the words or phrases. If the

English grammar states that pronominal objects follow the verb and the French one says that they precede, the same transfer grammar, though still without any explicit mention of order, will cause the appropriate "reordering" to take place. Similarly, nothing more would be required in the transfer grammar in order to place adjectives properly with respect to the nouns they modify, and so forth.

B. Semantics

It may be objected to the line of argument that I have been persuing that it requires the legs of the translating machine to be tied together at too lower a level, essentially at the level of syntax. To be sure, it allows more elaborate transfer grammars than the one just illustrated so that the translation of a sentence would not have to be structurally isomorphic with its source, *modulo* ordering. But the device is essentially syntactic. However, the relations that can be characterized by FUG and similar monotonic devices are in fact a great deal more diverse than this suggests. In particular, much of what falls under the umbrella of semantics in modern linguistics also fits conveniently within this framework. Something of the flavor of this can be captured from the following example. Suppose that the lexical entries for the words *all* and *dogs* are as follows:

$$\begin{bmatrix} \text{Cat} = \text{Det} \\ \text{Lex} = \text{all} \\ \text{Num} = \text{Plur} \\ \text{Def} = + \\ \text{Sense} = \begin{bmatrix} \text{Type} = \text{all} \\ \text{Prop} = \begin{bmatrix} \text{Type} = \text{Implies} \\ \text{P1} = [\text{Arg} = \langle \text{Sense Var} \rangle] \\ \text{P2} = [\text{Arg} = \langle \text{Sense Var} \rangle] \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \text{Cat} = \text{N} \\ \text{Lex} = \text{dog} \\ \text{Art} = \begin{bmatrix} \text{Num} = \text{Plur} \\ \text{Sense} = \langle \text{Sense} \rangle \end{bmatrix} \\ \text{Sense} = \begin{bmatrix} \text{Prop} = \begin{bmatrix} \text{P1} = \begin{bmatrix} \text{Type} = \text{Pred} \\ \text{Pred} = \text{dog} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

When the first of these is unified with the value of the *Art* attribute in the second as required by the grammar, the result is as follows:

$$\begin{bmatrix} \text{Cat} = \text{N} \\ \text{Lex} = \text{dog} \\ \text{Art} = \begin{bmatrix} \text{Cat} = \text{Det} \\ \text{Lex} = \text{All} \\ \text{Def} = + \\ \text{Num} = \text{Plur} \\ \text{Sense} = \langle \text{Sense} \rangle \end{bmatrix} \\ \text{Sense} = \begin{bmatrix} \text{Type} = \text{All} \\ \text{Prop} = \begin{bmatrix} \text{Type} = \text{Implies} \\ \text{P1} = \begin{bmatrix} \text{Type} = \text{Pred} \\ \text{Pred} = \text{dog} \\ \text{Arg} = \langle \text{Sense Var} \rangle \end{bmatrix} \\ \text{P2} = [\text{Arg} = \langle \text{Sense Var} \rangle] \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

This, in turn, is readily interpretable as a description of the logical expression

$$\forall q.dog(q) \wedge P(q)$$

It remains to provide verbs with a sense that provides a suitable value for P, that is, for ⟨Sense Prop P2 Pred⟩. An example would be the following:

$$\begin{bmatrix} \text{Cat} = \text{V} \\ \text{Lex} = \text{barks} \\ \text{Tense} = \text{Pres} \\ \text{Subj} = \begin{bmatrix} \text{Pers} = 3 \\ \text{Num} = \text{Sing} \\ \text{Anim} = + \end{bmatrix} \\ \text{Obj} = \text{NONE} \\ \text{Sense} = [\text{Prop} = [\text{P2} = [\text{Pred} = \text{bark}]]] \end{bmatrix}$$

IV Conclusion

It has not been possible in this paper to give more than an impression of how an experimental machine translation system might be constructed based on FUG. I hope, however, that it has been possible to convey something of the value of monotonic systems for this purpose. Implementing FUG in an efficient way requires skill and a variety of little known techniques. However, the programs, though subtle, are not large and, once written, they provide the grammarian and lexicographer with an emmense wealth of expressive devices. Any system implemented strictly within this framework will be reversible in the sense that, if it translates from language A to language B the, to the same extent, it translates from B to A. If the set S is among the translations it delivers for a, then a will be among the translations of each member of S. I know of no system that comes close to providing these advantages and I know of no facility provided for in any system proposed hitherto that it not subsumable under FUG