

Re: Universal grammar

Source: <http://sci.tech--archive.net/Archive/sci.lang/2006-10/msg01342.html>

- *From:* haberg@xxxxxxxxxx (Hans Aberg)
 - *Date:* Thu, 19 Oct 2006 12:23:46 GMT
-

In article <1161246509.794397.244600@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>, "Rob Freeman" <groups@xxxxxxxxxxxxxxxxxxxx> wrote:

I am interested in the parallel you get if you think of a "statement", or a rule, as an ordering over a set. If ordering A is true (that is, a valid ordering over the set), but incompatible ordering B is also true, then A and not A can be true, depending on the truth of B.

This happens all the time in the real world. For instance if you try to order people w.r.t. height and IQ at the same time.

This does not have anything to do with the Goedel theorems, which arise, roughly, by the assumption that there are infinities that can be treated as objects.

What I want to know is what this says about our ability to resolve a distribution down to some parametrization in terms of rules. Is it always possible to find a consistent set of rules which completely describes all the patterns in a distribution?

This is somewhat too diffuse to be possible to be pinned down in mathematics (see below, though). You might try the newsgroups comp.theory and comp.ai. Particular the idea how humans learn, and trying to emulate that in computers seem to be an AI problem – which means it is hard.

This is relevant to language, because it is exactly the problem you get if you try to learn natural language grammar from distributions of word associations.

In fact I have read it was inconsistencies of this kind which first led Chomsky to reject the idea language could be learned from examples and propose it must be parametrized innately. A "grammar" which because innate, must be universal. Hence a Universal Grammar, of which the important property was that it could not be observed in distributions

Re: Universal grammar

of word associations.

I think one knows pretty well that although the ability to learn languages is innate, languages itself are not, as there have been a few cases of children not getting the chance of learning it early in life, and when they grow up, they cannot learn it, or very little. If language somehow was innate, one should have it with the genes. So language is learned, and one is probably learning a collection of structural patterns, just as with all else.

In fact, the definition of a formal grammar closely follows this. (The description I use is in a book by Waite & Goos, "Compiler Construction", chapter 5, but it rather hard core math. And learning about parsers, there is the newsgroup comp.compilers – but it does not deal much about parsing of natural languages. In the Unicode List, "John D. Burger" <john AT mitre DOT org> mentioned he had been working with natural language processing for two decades, so perhaps he can give some more inputs.)

Anyway, a formal language L is a subset of the set V^* , the set of finite sequences (or strings) of a set V called the vocabulary (mathematicians call this the free monoid on the set V). Each member of L is called a sentence. So already here, one can define a language by just writing down all legal sentences. (A formal sentence could though be anything legal in the language, like a whole book, if one so sets the language definition.) The grammar comes into play only by the attempt to find a more structured description of L . The problem between learning a language naturally and finding a formal description of it arises from the inability to read off how the human brain structures the information it contains.

Then in order to pin down some types of languages, one uses a grammar. A production is a pair (s, p) , where s, p in V^* , and is written $s \rightarrow p$ (and some write it $s \leftarrow p$ or $s: p$). Now, if a string m in V^* contains s as a substring, it can be replaced by p to get a new string n , and one then says that m is directly rewritable into n by the rule $s \rightarrow p$. And if one has a set P of productions, and a string m is directly rewritable by a sequence of members of P into n , then one says that m is rewritable into n by P .

A formal grammar G is then a quadruple (T, N, P, Z) , where T is the set of terminals (or constants), corresponding to the words in a language L , N is the set of non-terminals, which are grammar symbols (or variables) like "noun", "verb", etc. in natural languages. The vocabulary V is the disjoint union of T and N . P is a set of productions on V . And Z is the sentence (or start) symbol. The language $L(G)$ derived as a function of the grammar G is then the set of members of T^* that can be derived by rewriting Z in V^* using the set of productions P .

So, from this, you can see that it is quite easy to define productions and add them to get a larger language, which answers part of your question. The problem, though, is that your language might become easily too large,

Re: Universal grammar

if your rules are too unspecific. So there, the theory has the problem that one has to play around with the extra grammar variables N , to get exactly the right language.

Now, the type of grammar used the most in implementing computer languages are the so called context free grammars (Chomsky hierarchy type 1), where each production (or rule) in P , has the form $A \rightarrow x$, where A in N , and x in V^* . This does not admit implementing say the construction of definitions, widely used in computer languages:

I could decide to introduce, in my computer language, the construct

```
define <name> ...
```

and then <name> will subsequently behave syntactically according to this definition. In a natural language parsing computer language, might have the construct:

```
define flies := noun.
```

and then this word should behave like that grammatically. Such constructs can be handled using so called attribute grammars. But a parser attaches actions to the productions, so one way is to make a table with the name "flies", and whenever this word appears in the input, the lexer returns the grammar variable "noun" to the parser.

Now, in an unambiguous language, once "flies" has been defined, either a new definition of it must be illegal, or override the old definition, say in a limited scope $\{...\}$ which is often called an "environment" in computer lingo. But in a natural language, it is legal to have

```
define flies := verb
```

admitting the ambiguous "Time flies like an arrow". This leads to a parser parsing several possibilities in parallel.

So, this answers in part your question (I hope). One can admit adding rules, but there will be ambiguities, and if the rules added are too narrow, the grammar will admit sentences which are not legal sentences in the language one wants to capture.

—

Hans Aberg

.