

Re: Universal grammar

Source: <http://sci.tech--archive.net/Archive/sci.lang/2006-10/msg01491.html>

- *From:* haberg@xxxxxxxx (Hans Aberg)
 - *Date:* Sat, 21 Oct 2006 14:12:11 GMT
-

In article <1161418795.730995.77650@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>, "Franz Gnaedinger" <frgn@xxxxxxxx> wrote:

Mathematical logic is a techno–logic, concerned with building, constructing, maintaining: www.seshat.ch/home/equal.htm

There is no problem dealing with that, as in metamathematics, one introduces the notion of a theory with equality.

And here you are with the universal grammar of Rupert Ruhstaller, based on functors and arguments, visualized in budding circles: www.seshat.ch/home/grammar.htm

This grammar will hardly help you, so let me introduce my grammar of equations from 1974/75. Every basic sentence can be given as an overlapping of three sets, one of them belonging to the word be, or it can be given as an equation, or it can be given as an object:

overlapping sets: ball be round
sentence: The ball is round.
equation: the ball – is – a round one
object: the round ball

overlapping sets: ball be roll
sentence: The ball rolls.
equation: the ball – is – a rolling one
object: the rolling ball

You recognize an object you could touch and hold: ball. The object is for real: be. You notice a smooth movement along a slight slope, at the same time a revolving movement of the object: roll. The ball, the being, and the rolling occur at once, at the same place, in the same object. The sets overlap, the equation is formed, the object named: the ball is rolling, the rolling ball – also the being of the rolling ball,

Re: Universal grammar

the rolling of the being ball, whatever comes first to your attention.

Perhaps you might want to use ASTs (abstract syntax trees – see Wikipedia), in order to get language independent representations. For example

The ball is round.

might be represented as the tree

```
  is
 /  \
the  round
 /
ball
```

That is, "is" acts as an operator with two arguments. The first argument has what computer scientists call a closure:

```
the
 |
ball
```

made up of a (linguistic) quantifier and a noun. And similarly for the second argument.

Most sentences are conglomerates of explicit and implicit equations.

The generalization is to deal with logical equations. In computer science, the first stepping stone are Prolog like programs. These handle equations without bound variables (as quantifiers "all", "exist", the Church "lambda" = function mapping " λ ", etc.) and is doing only a first depth search. The latter means that when attempting to prove the statement X by searching a clause

$A :- A_0, \dots, A_k$

corresponding to a provability statement

$A_k, \dots, A_0 \vdash A$

(if A_k, \dots, A_0 are valid object formulas, then so is A), it first uses a unification $u(X, A)$ as to obtain a valid substitution s, and adds

$s(A_0), \dots, s(A_k)$

as new conditions to prove, but the search is then only done sequentially. This way, one will miss valid proofs if there is an infinite branch in the proof, as not proof possibilities are searched.

So when turning a Prolog program into a proof checker, there are some things one must do:

- Add operators that bind variables – very difficult to get it right in practical implementation.
- Add breadth first search of the proof tree.
- Admit unification branching (get a sequence of substitutions from $u(X, Y)$).

Re: Universal grammar

When he beheld his shadow in the brook
The fishes spread on it their golden gills

He – is – Adonis, or perhaps Shakespeare himself
Shakespeare or Adonis – was – making an observation
what? – was – observed or beheld by him
the beheld – was – his own shadow in the brook
fishes – were – swimming through the shadow
the fishes in the shadow – were – hardly visible
visible – were – their gills
the gills – were – reflecting a light and thus shining
golden, although the fishes themselves were hardly
visible in the shadow cast by the poet or his alter
ego in the poem

An elementary school teacher was pleased with my method of dealing with sentences. The son of a boss of mine had a problem grasping the concept of mathematical equations while being good at language. I told him: forget about numbers and give me a sentence. He invented a sentence and I turned it into a series of linguistic equations. Then we did many more sentences together, for nearly two hours. He got the knack of it, and in the end he lost at least a part of his fear of mathematical equations.

Now my proposition for you, Hans. When you drive a car you follow a street, you hardly cruise across meadows and bushes. Prepare also a "street" for your program: by turning a mathematical text into basic equations.

In addition, math is structured into what corresponds to what computer scientists call environments (constructs like {...}) and modules (larger logical units).

In reality, working math is not dealing with a single metamathematical object theory, but a sequence of them. In addition, it is easy for a human to jump out of any limited metamathematical structure. There are a number of interesting problems.

—
Hans Aberg

.