UNDEE-AP'NEAN AND IMPROVING LISP PROGRAMS

Herald WERTZ
Laboratoire d'I, P. I. R. S. P.
75012 PARIS

We are currently investigating the creation of an
academic environment for the teaching of a programming
language to approximately 1800 students each year. At
this effect we are constructing an instructor and pupil
programming system: VISION & CAN & PHENARETE (GREUSAY
1977, GODDESS M. 1977, WERTZ 1976), a system designed
to help the individual apprentice in the process of de
developing, writing and debugging programs.

In this note we will describe some aspects of
PHENARETE, the concival part of the system: PHEN-
ARETE receives as input a student proposition of a program
- which may contain as well syntactic as semantic errors
- and delivers as output two improved propositions of
this program: a finite sequence of approximations.

With the help of our system, the learning of a
programming language is done in a cycle:

<table>
<thead>
<tr>
<th>Improvement</th>
<th>Proportion</th>
<th>Approximation</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>Proposition</td>
<td>System PHENARETE</td>
</tr>
</tbody>
</table>

Before constructing PHENARETE we observed for
one year our students beginning to learn LISP, to see
which kinds of errors are statically the most current.
There we found principally five sorts of errors:
- absence of variables
- inversion of variables
- grouping errors
- absence of the conditional instruction
- non-termination of a computation.

Our system is particularly efficient in the detection of
these kinds of errors. It doesn't try to verify the program
with respect to the axioms of the programming, but it
tries to understand the programming language constructs
used in the program, to see the interaction of the dif
erent parts of the program and to detect inconsistencies,
with a particular emphasis on the verification of the
termination of the program. In some way PHENARETE
may be considered as an intelligent Interpreter: she does
do execute but she interprets the program, and the result
of an interpretation are some propositions (propositional
interpretations). PHENARETE incorporates the programming knowledge of a programming apprentice.

Let me illustrate the reasoning involved in an inter
pretation of a sample LISP program (main step only):
assume a student has submitted the following program
(the numbers are for references in the text).

1. (DEF REV (X Y) (REIREV X (CONS (CAR X) Y)))
2. (DEF REV X (CONS (CAR X) Y))
3. (值班 Y) XI)

First, lines 2 and 3) the grouping of the body of
the function suggests that the user has omitted the
COND--function call, so PHENARETE introduces (line
2-6) (COND). Then (line 2) we find as the first clause
of the COND immediately a recursive call of the function
REV, and following line 3 an another clause. Knowing that
line 3 can never be attained during an execution of the
program we can invert the two clauses. This gives:

1'. (DEF REV X Y)
2'. (值班 X)
3'. (值班 Y) XI)
4'. (值班 X)

So far the surface improvements. Now a closer
look at line 4' tells us that X and Y will be lists, when
the function is invoked, and that the first argument (X)
is unchanged and the second argument (Y) will grow
longer. In line 3' the case second argument = NIL is
isolated, but this is the only case where the computation
of REV will terminate. If Y is different of NIL, the
only constructive computation done in the program is
the creation of the new list argument 2. So, when hypo-
thesizing that the result of REV will be this list just
created, we have to form a recursion step. Let us try to
call REV with the CDR of argument 3:

1' (值班 X CDR X) (CONS (CAR X) Y))

but the recursion will not stop either. So let's try to
introduce a supplementary test (值班 X). And, always
under the hypothesis that Y will be the result of the
computation, this gives us:

1 (值班 X) Y1.

This will be the first proposition of PHENARETE:

Proposition 1

1. (DEF REV X Y)
2. (值班 X)
3. (值班 Y) XI)
4. (值班 X)

But this new line is just the line 3' with X and Y
inversed, so let's try without line 2. Ok, that is also a
recursive procedure which stops with the correct algorithm,
as another proposition:

1. (DEF REV X Y)
2. (值班 X)
3. (值班 Y) XI)
4. (值班 X)

Finding no more possible interpretation, PHENARETE
will stop, after these two proposition. In the hypo-
thesis phase she justifies all the modifications she proposes
in a way similar we did.

The system is implemented in VLISP-10 and is
currently used by about 1900 students.

REFERENCES:
CHAILLOUX J., VLISP 10, Manuel de references, Dpt. d'Informatique,
GREUSAY, Differences d'interpretation de systemes VISION,
WERTZ H., PHENARETE, Springer-Verlag, G. B. Jahrhundert, Stuttgart,
Oct-1976, 44-441.