Formal Semantics of ALGOL 60: a comparison of four descriptions

Cliff B. Jones
Troy K. Astarte
Newcastle University
Why define a language?

- Programmer
- Natural language
- Compiler writer
- Program
- Language designer
- Machine language
- Computer 1
- Computer 2
What is ‘formal semantics’?

- Formal: rigorous, mathematical, ‘tractable’
- ‘Meaning’?
  - Procedural programming languages
  - The ability to reason about the effect of a program
- Two centres influencing / competing:
  - IBM Laboratory, Vienna: Zemanek, Lucas, …, Jones
  - PRG, Oxford: Strachey, Scott, Wadsworth
Why ALGOL 60?

❖ “a language so far ahead of its time, that it was not only an improvement on its predecessors, but also on nearly all its successors.” — Tony Hoare

❖ An interesting history: design by an IFIP Working Group

❖ *The* language of academia when semantics emerged

❖ Many features: nested phrases; jumps; recursion

❖ Many different descriptions & fragments

❖ ALGOL Reports—BNF; but informal semantics
Example

begin real procedure A(k, x1, x2, x3, x4, x5);
  value k; integer k;
  begin
    real procedure B;
    begin
      k := k - 1;
      B := A := A(k, B, x1, x2, x3, x4)
    end;
    if k < 0 then A := x4 + x5 else B
  end;
end;

outreal(A(10, 1, -1, -1, 1, 0))

end;
Formal Language Description Languages

- September 1964, Baden-bei-Wien, IFIP TC2 organised
- Many semantic ideas on show
- Strachey presents informal precursor of denotational semantics
- No relevant Vienna speaker; but McCarthy inspires operational —and abstract syntax
- Landin presents a formal mapping to IAEs
- F. G. Duncan: ultimate metalanguage
- Most speakers do not go on to work in semantics
VDL operational description (1968)

- PL/I: a language to replace FORTRAN & COBOL
- IBM Vienna takes on PL/I language description in 1964
- LDH/LDV show increasing formalism in correspondence 1964–1965
- Known as Vienna Definition Language outside IBM
- Zemanek wants to demonstrate VDL (ULD-IIIvII) technique on smaller language
- ALGOL 60 description authored by logician Peter E. Lauer
Exit operational description (1972)

- Cliff Jones on assignment in 1968; returned 1970
- Twin machine paper: using formal definition in language design.
- Difficult lemma!
- Alternative jump handling: exit mechanism (1970)
- Error checking still dynamic, but translator notes
- ALGOL 60 definition authored by Dave Allen, Dave Chapman, & Cliff Jones
Oxford denotational description (1974)

- Penrose had suggested λ-calc to Strachey in 1958
- Strachey meets Scott at WG2.2 Vienna meeting (Apr 1969)
- Scott in Oxford for autumn 1969; solves problem in λ-calc
- Mathematical semantics: smaller state, greater abstraction
- Jumps tricky: continuations (Wadsworth)
- ALGOL 60 definition authored by Peter D. Mosses
- Work during PhD; thesis in 1975 (on SIS)
VDM denotational description (1978)

- Jones back in Vienna in 1973
- FS project included PL/I compiler… formalised!
- Definition in 1974, denotational approach with exit mechanism
- FS killed, but Jones & Bjørner salvaged ‘VDM’
- ALGOL definition as a demonstration of concept
- ALGOL 60 definition authored by Cliff Jones & Wolfgang Henhapl (republished 1982)
- Equal abstraction to Mosses, but more readable
Different semantic approaches

- Operational vs. denotational vs. axiomatic [not covered]
- (interpreting) (mapping to fns) (giving properties)
- Notion of ‘state’: store, or more?
- Syntax: concrete or abstract?
- Error handling: static or dynamic?
- “check those things which rely only on symbol matching and omit those checks which, in general, rely on values of symbols” — ACJ
Our paper

- Still in draft!
- Intro: why semantics, early semantics, ALGOL, & Report
- For each definition (presented chronologically):
  - Historical notes & context
  - Version of ALGOL
  - Syntactic issues
  - Overall semantic style
  - Specific points: jumps, procedures, environment/state
- Conclusions: some comparisons & other significant descriptions.
Timeline

1960
- FORTRAN (VDL)

1970
- BASIC (VDL)
- ACJ (VDM)

1980
- PL/I (VDM)
- PDM (VDM)
- CHILL (VDM)

1990
- HJ (VDM)
- HJ2 (VDM)
- Prolog (VDM/VDL)
- Modula-II (VDM)