EQUIVALENCE OF DENOTATIONAL AND OPERATIONAL SEMANTICS

Since the denotational and operational styles of semantics are, in some sense, alternatives to each other, it would be good to show that they are equivalent. i.e. that one form of semantic definition can be transformed into the other with no loss of meaning.

Both operational and denotational semantics are based on transformations of store as it is modified by successive commands. In the operational style, the transformations are expressed in a relation. Thus

\[ [c, \sigma] \rightarrow \sigma' \]

Says that command \( c \) executed in some store, results in a changed store. Clearly the denotational style is similar in spirit:

\[ C[s] = s' \]

But this is expressed as a function. If we are to show that the two semantics are equivalent, then we need to show that

\[ C[[C]] = \{ [s, s'] | [C, s] \rightarrow s' \} \]

Equivalently we can express this as

1. \( [s, s'] \in C[[C]] \Leftrightarrow [C, s] \rightarrow s' \)

Since the different forms of the commands include expressions, both arithmetic and boolean, we also have the evaluation of these:

2. \( E[[E]] = \{ [s, n] | [E, s] \rightarrow n \} \)

And

3. \( B[[B]] = \{ [s, t] | [B, s] \rightarrow t \} \)

Since equation 1 relies on equations 2 and 3, these must be proved first. Both can be proved by structural induction on the form of the expression. The induction hypothesis for 2 is

\[ P(E) \Leftrightarrow E[[E]] = \{ [s, n] | [E, s] \rightarrow n \} \]

and the hypothesis for 3 is

\[ P(B) \Leftrightarrow B[[B]] = \{ [s, t] | [B, s] \rightarrow t \} \]

The proofs of the simple forms, when the expression is an identifier or a constant are obvious, and the proof of the arithmetic or relational expressions are done by considering the form of their derivations in the operational semantics.

To prove 1, we split it into two proofs, the forward implication and then the reverse. To prove

\[ [s, s'] \in C[[C]] \Rightarrow [C, s] \rightarrow s' \]

we proceed by using rule induction on the form of each command. To prove that

\[ [C, s] \rightarrow s' \Rightarrow [s, s'] \in C[[C]] \]

we proceed by structural induction on the form of commands, with a use of normal mathematical induction for the while loop.

The two styles of semantics are thus equivalent.