A Proof of the Rule of Consequence in Inference
(Programming Language)

By Myoung Lyoul Kim

Jeonbug National University, Jeonju, Korea

1. Introduction

Recent years have witnessed major advances both in the semantics of programming language, and in the methodology of program design and verification.

The rule of consequence comes from the soundness of inference which is a part of proof theory on the basis of the concept and semantics.

Addition to this, rule of composition and rule of conditionals consist proof rules.

We introduce three classes of constructs:

- lexp (for integer expressions) with typical elements s, t, ..., 
- boolp (for boolean expressions) with typical elements b, ..., and 
- Stat (for statements) with typical elements S, ...

And we have two classes of semantic entities:

- Ivar (for integer variables) with typical elements x, y, z, u, ..., and 
- Icon (for integer constants) with typical elements m, n, ...

V is used to denote the set of integers, with typical elements α, ..., 
B is used to denote the set of truth-values {tt, ff}, with typical elements β, ..., 
Σ is used to denote the set of states, i.e., of all functions from Ivar to V, and has typical elements ω, ...

We define two functions as follows.

\[ T : Assn \rightarrow (\Sigma \rightarrow B) \]
\[ F : Form \rightarrow (\Sigma \rightarrow B) \]

2. Definitions

(1) An inference is a construct of the form \( \frac{f_1, \ldots, f_n}{f} \), with \( f_i, i=1, \ldots, n, \) and \( f \in Form \).

(2) An inference \( \frac{f_1, \ldots, f_n}{f} \) is called sound, written as

\[ \frac{f_1, \ldots, f_n}{f} \]

whenever validity of \( f_i, i=1, \ldots, n, \) implies validity of \( f \).
3. **Theorem** (rule of consequence)

\[
\begin{array}{c}
(p \Rightarrow p_1), \{p_1\} S(q_1), (q_1 \Rightarrow q) \\
\{p\} S(q)
\end{array}
\]

**Proof.** Assume that for all \( \sigma \), \( T(p)(\sigma) \Rightarrow T(p_1)(\sigma) \)
and, for all \( \sigma \), \( T(p_1)(\sigma) \Rightarrow T(q_1)(M(S)(\sigma)) \),
and, for all \( \sigma \), \( T(q_1)(\sigma) \Rightarrow T(q)(\sigma) \).
Now, we take \( \sigma' = M(S)(\sigma) \).
Then \( T(p)(\sigma') \Rightarrow T(p_1)(\sigma') \),
\( T(p_1)(\sigma') \Rightarrow T(q_1)(\sigma') \)
and, \( T(q_1)(\sigma') \Rightarrow T(q)(\sigma') \),
so that \( T(p)(\sigma') \Rightarrow T(q)(\sigma') \).
Hence \( T(p)(\sigma) \Rightarrow T(q)(M(S)(\sigma)) \).

References

