How to use Hoare Logic to prove program correctness

Junshu Xing
231220030@smail.nju.edu.cn
Nanjing Univesrity
Nan Jing Jiang Su, China

Abstract

Hoare Logic is a widely used logical system for program verification, applied to reason about imperative language programs. Its fundamental concept involves constructing a contractual specification between the code and its callers, consisting of a precondition and a post- condition. Both the precondition and post-condition are assertions used to describe the program's specification. The precondition outlines the conditions that must be satisfied in the program's state before code execution, encompassing aspects such as parameter value ranges and data structures. Meanwhile, the post-condition details the conditions that need to be satisfied in the program's state after correct execution, primarily involving relevant information about parameters post-execution.

ACM Reference Format:
Junshu Xing. 2023. How to use Hoare Logic to prove program correctness. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 The basic idea of Hoare Logic

When verifying a program using Hoare Logic, the correctness of the program is represented by a Hoare triple P S Q, where P is the precondition, S is the program segment, and Q is the post-condition. We will talk about it later.

2 The fundamental principles of Hoare Logic

Hoare Logic is based on the formal proof of program correctness through the insertion of assertions into the program. It provides a foundational approach to verify the correctness of programs, offering specific reasoning rules for fundamental control structures in high-level languages, such as conditional statements and loop statements. To prove the correctness of a program, one needs to continuously apply these reasoning rules for deduction. These rules primarily involve the following:

2.1 skip principle

skip:{P}skip{P}

Formula (1) is the Skip Rule, indicating that if P holds before the execution of a skip statement (an empty statement), it continues to hold true after the execution. The execution of an empty statement does not alter the value of P.

2.2 assign principle

assign:{P[x↦a]}:=a{P}

Formula (2) represents the Assignment Axiom, where P[x↦a] denotes replacing occurrences of x in P with a. The axiom states that if P is true after the execution of the assignment statement x = a, then P[x↦a] is true before the assignment, where x is a free variable.

2.3 sequence principle

sequence: $\frac{\{P\}S_1\{Q\},\{Q\}S_2\{R\}}{\{P\}S_1;S_2\{Q\}}$

Formula (3) is the Composition Rule, stating that P S1; S2 R holds if P S1 Q and Q S2 R are both true.

2.4 if principle

if: $\frac{\{b\wedge P\}S_1\{Q\},\{\neg b\wedge P\}S_2\{Q\}}{\{P\}if(b)then(S_1)else(S_2)\{Q\}}$

Formula (4) is the Conditional Rule, representing that if S1 is executed, then P and b are true before execution, and if S2 is executed, then P is true, and b is false before execution.

2.5 while principle

while: $\frac{\{b\wedge P\}S\{P\}}{\{P\}while(b)do(S)\{\neg b\wedge P\}}$

Formula (5) is the Loop Rule, indicating that if P and b are true, execute S. After the execution of S, b is false, and P is true. P represents the loop invariant that must be maintained as true throughout the loop.

2.6 cons principle

cons: $\frac{P\rightarrow P_1,\{P_1\}S\{Q_1\},Q\rightarrow Q_1}{\{P\}S\{Q\}}$

Formula (6) is the Consequence Rule, expressing a causal relationship. If a Hoare triple is true, strengthening the precondition or weakening the post-condition also holds true.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

3 The limitations of Hoare Logic

Hoare logic is currently widely used as a classic method for proving properties of imperative language programs. Although it can be easily extended to determine some simpler aliasing relationships, it struggles with handling complex aliasing relationships in pointer programs.Hoare logic also encounters difficulties when it comes to verifying programs involving manipulable data structures. Consequently, research on Separation Logic has emerged to address this challenge.