Formalization of Asymptotic Notations in HOL4

Nadeem Iqbal, Osman Hasan, Umair Siddique, Falah Awwad

National University of Sciences and Technology, Islamabad, Pakistan
College of Engineering, UAE University, Al Ain, UAE

ICCCS 2019, Singapore
February 25, 2019
Outline

• Introduction

• Proposed Methodology

• Formalization Definitions

• Formal Verification

• Conclusions
Asymptotic Notations

• Used for computational time assessment of Algorithms

• The asymptotic notation based assessment is independent of
  • Language
  • Execution Platform
  • Compiler
  • Input data
Asymptotic Notations

• **Big-\(O\) notation or simply \(O\)-notation was introduced by a number theorist **Bachmann in 1894**

• **Little-\(o\) notation was introduced by **Landau in 1909**

• **Big-\(\Omega\), Big-\(\Theta\), and Little-\(\omega\) notations were presented by **Knuth in 1976**
Types of Analysis

- Runtime Complexity Analysis
  - Traditional
    - Paper Pencil Methods
  - Formal Methods
    - Simulations
    - Model Checking
    - Theorem Proving
Comparison of Analysis Techniques

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Paper-and-Pencil Proof</th>
<th>Simulation</th>
<th>Model Checking</th>
<th>Higher-order-logic Proof Assistants</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expressiveness</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Accuracy</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Automation</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
</tbody>
</table>

Given the extensive usage of asymptotic analysis of algorithms in safety-critical systems, there is a dire need of using formal methods support in this domain.
Contributions of this paper

• A library of formalized asymptotic notations, i.e., $O$, $\Theta$, $\Omega$, $o$ and $\omega$, in the higher-order-logic theorem prover HOL4 using the real number theory
  • Formal Definitions of $O$, $\Theta$, $\Omega$, $o$ and $\omega$ in HOL4
  • Formal Verification of Properties of $O$, $\Theta$, $\Omega$, $o$ and $\omega$ in HOL4
HOL4 Theorem Prover

- Higher-order-logic Proof Assistant
  - Notation: ML
  - Small Core:
    - 5 basic axioms
    - 8 primitive inference rules
- Numerous automatic proof procedures are available
- Supports Reasoning about
  - Real Numbers
  - Calculus
Proposed Approach for Asymptotic Notations Based Analysis

HOL4 Theories
- Real Numbers
- Arithmetic
- Calculus

Formal Definitions of Notations

Formal Verification of Asymptotic notation Properties as Theorems

HOL4 Theorem Prover

Formally Verified Computational Complexity

Algorithm

Computational Complexity

Theorems
Formal Definitions: BigO

• The BigO takes a function $g$ as an input and returns the set of all functions $f$ which qualify the condition $0 \leq f(n) \leq c \cdot g(n)$.

Definition 1: BigO Notation

$$\forall g. \text{BigO} (g: \text{num} \rightarrow \text{real}) = \{(f: \text{num} \rightarrow \text{real}) | (\exists c \ n_0. (\forall n. n_0 \leq n \land 0 < c \implies 0 \leq f(n) \leq c \cdot g(n)))\}$$

• Here $f$ and $g$ are functions which take a natural number $\text{num}$ and return a real number $\text{real}$.

• The constants $c$ and $n_0$ are of type real and num, respectively.
Formal Definitions

**Definition 2: BigTheta Notation**
\[ \forall g. \text{BigTheta}(g:\text{num} \rightarrow \text{real}) = \{(f:\text{num} \rightarrow \text{real}) | \]
\[ (\exists c_1 c_2 n_0. (\forall n. n_0 \leq n \land 0 < c_1 \land 0 < c_2 \implies 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n))\} \]

**Definition 3: BigOmega Notation**
\[ \forall g. \text{BigOmega}(g:\text{num} \rightarrow \text{real}) = \{(f:\text{num} \rightarrow \text{real}) | \]
\[ (\exists c n_0. (\forall n. n_0 \leq n \land 0 < c \implies 0 \leq c g(n) \leq f(n))\} \]

**Definition 4: LittleO Notation**
\[ \forall g. \text{LittleO}(g:\text{num} \rightarrow \text{real}) = \{(f:\text{num} \rightarrow \text{real}) | \]
\[ (\exists c n_0. (\forall n. n_0 \leq n \land 0 < c \implies 0 \leq f(n) < c g(n))\} \]

**Definition 5: LittleOmega Notation**
\[ \forall g. \text{LittleOmega}(g:\text{num} \rightarrow \text{real}) = \{(f:\text{num} \rightarrow \text{real}) | \]
\[ (\exists c n_0. (\forall n. n_0 \leq n \land 0 < c \implies 0 \leq c g(n) < f(n))\} \]
Formal Verification

- Using the formal definitions of Asymptotic notations, we formally verified their properties
  - Transitivity
  - Symmetry
  - Transpose symmetry
  - Reflexivity

- using the HOL4 theorem prover

- The properties not only ensure the correctness of our definitions but also play a vital role in the formal complexity analysis of algorithms
Properties of O Notation

Theorem 1: Transitivity of O-Notation
\[ \forall f \ g \ h. \ f \in (\text{BigO} \ g) \land g \in (\text{BigO} \ h) \implies f \in (\text{BigO} \ h) \]

Theorem 2: Sum of O-Notation
\[ \forall t_1 \ t_2 \ g_1 \ g_2. \ t_1 \in (\text{BigO} \ g_1) \land t_2 \in (\text{BigO} \ g_2) \implies (t_1 \ n + t_2 \ n) \in (\text{BigO} (\text{max} (g_1 \ n, g_2 \ n))) \]
Conclusions

• Formalization of Asymtotic notations ($O$, $\Theta$, $\Omega$, $o$ and $\omega$) in HOL4
• Formal Framework for computational complexity analysis of algorithms

• Advantages
  • Accurate Results
  • Reduction in user-effort while formally Helpful in discovery of different pathways
Future Directions

• Applications in Cryptography
  • To estimate the size of the key so that it will be infeasible to break a system using given number of steps
  • Security assessment of authentication protocols, such as, the security proof of password authentication protocols
Thanks!

Questions