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

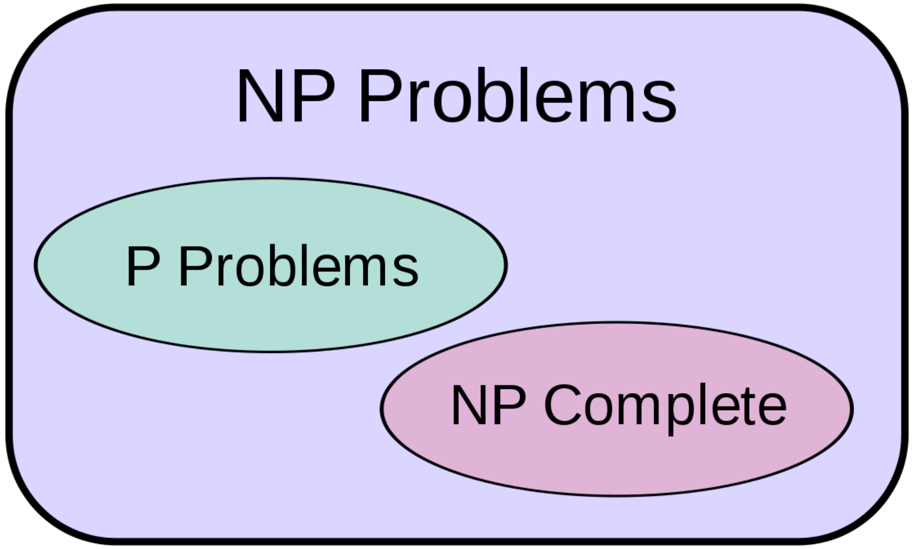
Outline

- Introduction
- Proposed Methodology
- Formalization Definitions
- Formal Verification
- Conclusions

Asymptotic Notations

- Used for computational time assessment of Algorithms

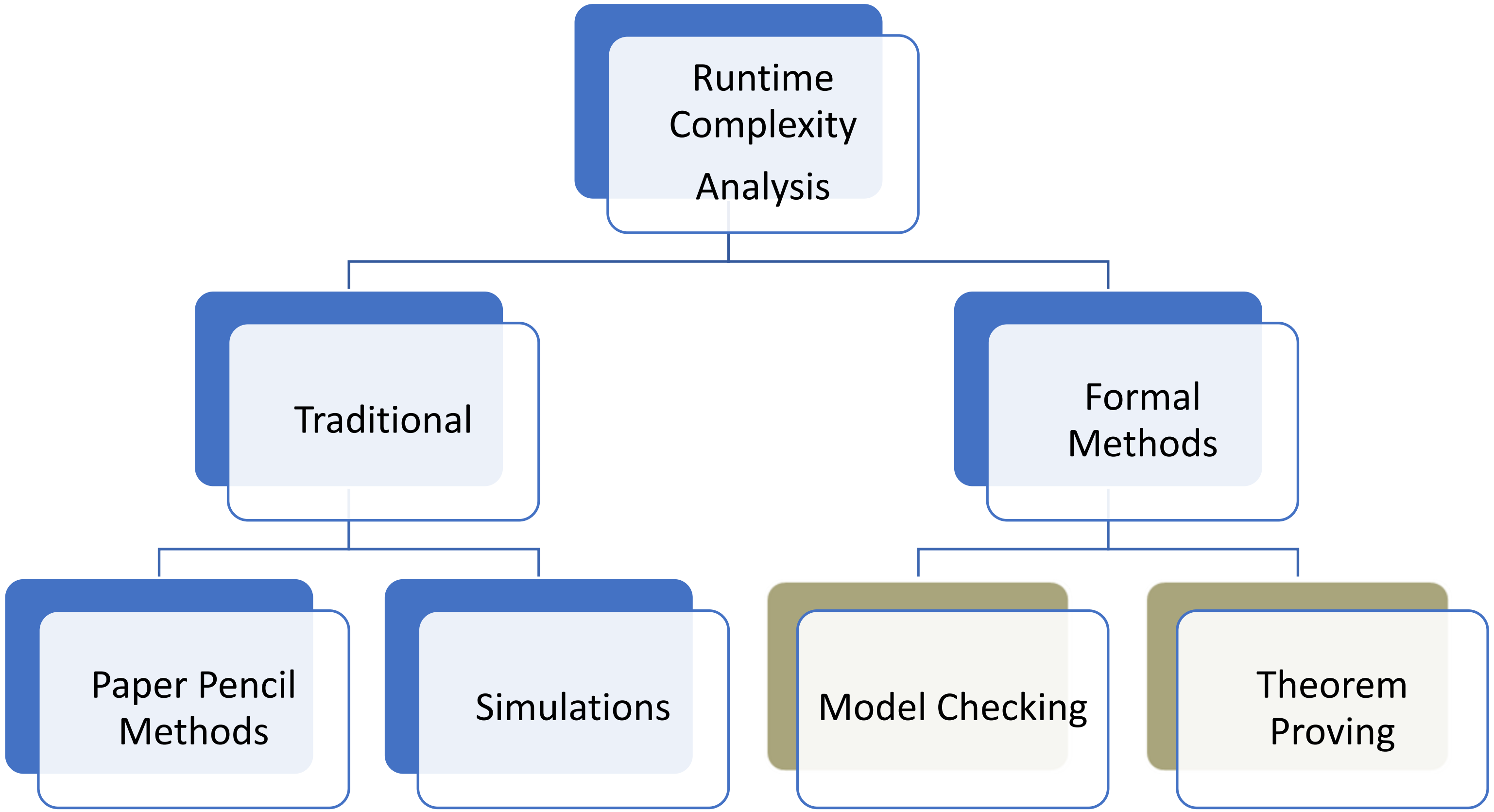
- The asymptotic notation based assessment is independent of
  - Language
  - Execution Platform
  - Compiler
  - Input data



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-Ω, Big-Θ, and Little-ω notations were presented by Knuth in 1976

Types of Analysis



Comparison of Analysis Techniques

Criteria	Paper-and-Pencil Proof	Simulation	Model Checking	Higher-order-logic Proof Assistants
Expressiveness	☑	☑	☒	☑
Accuracy	☒	☒	☑	☑
Automation	☒	☑	☑	☒

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

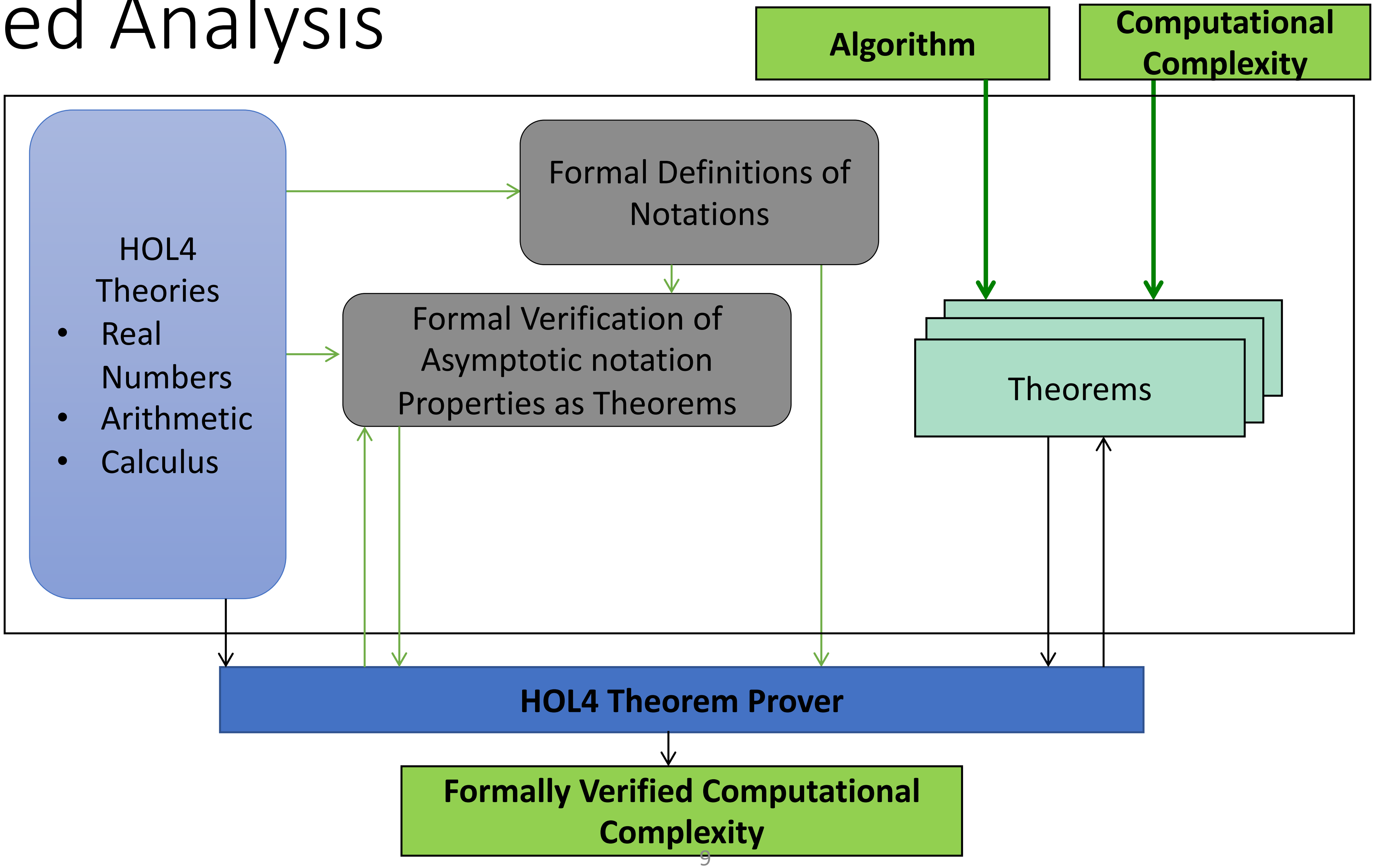
Contributions of this paper

- A library of formalized asymptotic notations, i.e., O, Θ, Ω, o and ω, in the higher-order-logic theorem prover HOL4 using the real number theory
  - Formal Definitions of O, Θ, Ω, o and ω *in HOL4*
  - Formal Verification of Properties of O, Θ, Ω, o and ω *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



Formal Definitions: BigO

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

Definition 1: *BigO Notation*

$$\vdash \forall\ g.\ \mathtt{BigO}\ (g:\mathtt{num} \rightarrow \mathtt{real}) = \{(f:\mathtt{num} \rightarrow \mathtt{real}) \mid (\exists\ c\ n\_0.(\forall\ n.\ n\_0 \leq n \land 0 < c \Longrightarrow 0 \leq f(n) \leq c\ g(n)))\}$$

- Here f and g are functions which take a natural number *num* and return a real number *real*
- The constants c and n_0 are of type real and num, respectively

Formal Definitions

Definition 2: *BigTheta Notation*

$$\vdash \forall\ g.\ \texttt{BigTheta}\ (g:num \rightarrow real) = \{(f:num \rightarrow real) \mid (\exists\ c1\ c2\ n\_0.(\forall\ n.n\_0 \leq n \wedge 0 < c1 \wedge 0 < c2 \Longrightarrow 0 \leq c1\ g(n) \leq f(n) \leq c2\ g(n)))\}$$

Definition 3: *BigOmega Notation*

$$\vdash \forall\ g.\ \texttt{BigOmega}\ (g:num \rightarrow real) = \{(f:num \rightarrow real) \mid (\exists\ c\ n\_0.(\forall\ n.\ n\_0 \leq n \wedge 0 < c \Longrightarrow 0 \leq c\ g(n) \leq f(n)))\}$$

Definition 4: *LittleO Notation*

$$\vdash \forall\ g.\ \texttt{LittleO}\ (g:num \rightarrow real) = \{(f:num \rightarrow real) \mid (\exists\ c\ n\_0.(\forall\ n.\ n\_0 \leq n \wedge 0 < c \Longrightarrow 0 \leq f(n) < c\ g(n)))\}$$

Definition 5: *LittleOmega Notation*

$$\vdash \forall\ g.\ \texttt{LittleOmega}\ (g:num \rightarrow real) = \{(f:num \rightarrow real) \mid (\exists\ c\ n\_0.(\forall\ n.\ n\_0 \leq n \wedge 0 < c \Longrightarrow 0 \leq c\ g(n) < f(n)))\}$$

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*

$\vdash \forall$ `f g h. f` $\in$ `(BigO g)` $\wedge$ `g` $\in$ `(BigO h)` $\Longrightarrow$ `f` $\in$ `(BigO h)`

Theorem 2: *Sum of O-Notation*

$\vdash \forall$ `t1 t2 g1 g2. t1` $\in$ `(BigO g1)` $\wedge$ `t2` $\in$ `(BigO g2)` $\Longrightarrow$
`(t1 n + t2 n)` $\in$ `(BigO (max (g1 n, g2 n)))`

Conclusions

- Formalization of Asymtotic notations (O, Θ, Ω, o and ω) 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