Conceptual Framework for Trustworthy Artificial Intelligence: Combining Large Language Models with Formal Logic Systems

Andrey Nechesov, Dmitry Kondratyev, Dmitry Sviridenko, Igor Anureev, Natalia Garanina, Andrei Gumirow, Ivan Gorobets & Yana Dementyeva

The Artificial Intelligence Research Center of Novosibirsk State University
1, Pirogova str., Novosibirsk, 630090, Russia

25 march 2025

Motivation

- Large Language Models (LLMs) increasingly used for problem-solving
- Need for reliable verification of generated solutions
- Challenge: Ensuring trustworthiness with reasonable resources
- Solution: Polynomial-time verification framework

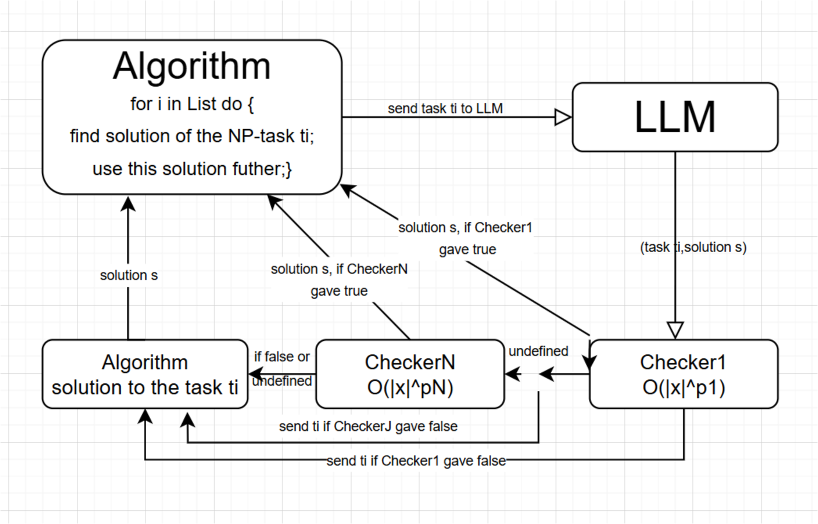
Core Concept

Theoretical Foundation

Idea: NP-complete problem solutions can be verified in polynomial time [1]

- The framework uses checkers to validate AI-based solutions
- Checkers implemented via **polynomial-time programming**
- Guaranteed polynomial-time complexity
- Tested on 2-SAT problems

Core Concept



Theoretical Definitions

Solver and Checker Environment

- Solver S: Function (possibly partial) $D \rightarrow R$
- Checker environment $E = (\Psi, \prec)$:
  - Ψ: Set of checker functions $\psi_j \in D_j \times R_j \rightarrow bool$
  - $\prec$: Partial order on Ψ

Environment E

The checking environment E of the solver S provides verification of the correctness of the output o of S in the input i, i.e. the correctness of the pair (i, o), by applying checkers from Ψ.

- The *true* value of the checker means that the solver returned the correct result o at input i,
- The *false* value does that the solver's result is incorrect at this input,
- And the undefined value $\perp$ does that the checker could not make an estimate.

Theoretical Definitions

Solver and Checker Environment

- Solver S: Function (possibly partial) $D \rightarrow R$
- Checker environment $E = (\Psi, \prec)$:
  - Ψ: Set of checker functions $\psi_j \in D_j \times R_j \rightarrow bool$
  - $\prec$: Partial order on Ψ

Reliability Condition

Solver S is reliable w.r.t. E on $I' \subseteq I$ if for each $i \in I'$ there exists a path $\psi_{m_1} \prec \ldots \prec \psi_{m_l}$ where:

- $\psi_{m_r}(i) = \perp$ for $1 \leq r \leq l-1$
- $\psi_{m_l} \in bool$

Conceptual Framework

The system

Conceptual framework $(S, E, \Delta_1, \Delta_2)$, where:

- (S, E): Solver with checker environment
- Δ_1: Checker verification tools. We use formal verification methods (in particular, the deductive verification method [2]) for ensuring the correctness of the checkers themselves).
- Δ_2: Checker complexity evaluation tools. We use polynomial-time programming methodology in Turing-complete languages [3] to check whether a checker corresponds to class P.

Conceptual Framework

Hybrid System Advantages

- S: Intelligent component (LLM)
- E: Analytical verification component
- Checkers often simpler to implement than solvers
- Reduces developer effort
- High operating speed

Case Study: 2-SAT Problems

Two Practical Applications

1. Heterogeneous Resource Allocation (HRA) [4]
2. Wireless Sensor Network Connectivity Analysis [5]

HRA Task

$$\begin{aligned} \mathscr{B} &= \{B_1, B_2, \ldots, B_n\} \text{ (blockchains)} \\ \mathscr{R} &= \{\text{GPU}, \text{CPU}\} \\ T(B_i) &: \text{Processing time for blockchain } B_i \\ B_j \prec B_k &: \text{Processing order constraint} \end{aligned}$$

Objective:

$$\min\left(\max_{r \in \mathscr{R}}\left(\sum_{B_i \text{ assigned to } r} T(B_i)\right)\right)$$

Case Study: 2-SAT Problems

Two Practical Applications

1. Heterogeneous Resource Allocation (HRA) [4]
2. Wireless Sensor Network Connectivity Analysis [5]

WSN Importance

- Critical for smart city applications [6]
- Communication represented as graph:
  - Vertices = sensors
  - Edges = direct communication links

Solvers

HRA_solver

WSN_solver

Conclusion

- Presented framework for reliable LLM solution verification
- Polynomial-time complexity guaranteed
- Reducing developer efforts by implementing only checkers instead of whole solvers
- Hybrid intelligent+analytical approach
- Practical applications demonstrated via 2-SAT problems
- Future work: Extension to other problem classes

References

- Marcelo Prates et al. (2019) Learning to solve NP-complete problems: A graph neural network for decision TSP
- Hähnle, Reiner and Huisman, Marieke. (2019) Deductive software verification: from pen-and-paper proofs to industrial tools
- Goncharov Sergey et al. (2024) Programming Methodology in Turing-Complete Languages
- Ait Aba et al. (2019) Heterogeneous resource allocation
- Biró & Kusper (2018) Wireless sensor network connectivity analysis
- Belghith & Obaidat (2016) Smart city applications
- Faye & Chaudet (2015) WSN importance