Karlsruhe Institute of Technology

LLMs in Formal Verification

Kickoff Event

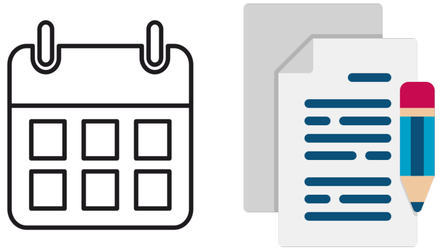
Chair: Application-oriented Formal Verification

Lecturers:

Prof. Bernhard Beckert
Debasmita Lohar
Michael Kirsten
Philipp Kern
Samuel Teuber
Jonas Schiffl
Jonas Klamroth
Mattias Ulbrich

April 30, 2025

Logistics



Course Registration

- Deadline: **May 11, 2025**

  Note: No registration/deregistration possible after this date!

- Instruction Language: **English**

Reading Assignment

- Read the assigned papers
- Find and read **1-2 additional papers** in the area (more is welcome but ***not*** required!)
- Build a foundation on the given topic

Meetings: Once in every 2 weeks (schedule appointment with your advisor)

Presentations / Discussion

Time — **15:45 - 17:15** (25 + 5 + 25 + 5 + 30 minutes), Room No. **236**

June 26:

presentation 1 Theorem Proving

presentation 2 Loop Invariant Synthesis

July 3:

presentation 3 Program Specification

presentation 4 Program Synthesis

July 17:

presentation 5 Bug Detection

presentation 6 Program Repair

July 24:

presentation 7 Natural Language Requirements

Presentations / Discussion

Time — **15:45 - 17:15** (25 + 5 + 25 + 5 + 30 minutes), Room No. **236**

June 26:

presentation 1 Theorem Proving

presentation 2 Loop Invariant Synthesis

July 3:

presentation 3 Program Specification

presentation 4 Program Synthesis

July 17:

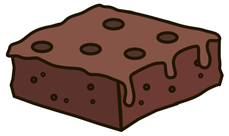
presentation 5 Bug Detection

presentation 6 Program Repair

July 24:

presentation 7 Natural Language Requirements

Discussion — Presenters are responsible for leading the discussion

Everyone must attend,  points for contributing to the discussion!

Feedback for Presenter: For future improvements! No impact on grades!

Writing Assignment

Report: 7-8 pages (strict), ACM Generic Journal Manuscript Format

Submission: **September 30, 2025**

- Topic: brief overview including
  - motivation,
  - different methods, their strengths and weaknesses,
  - discussion of results, and
  - conclusion

Writing Assignment

Report: 7-8 pages (strict), ACM Generic Journal Manuscript Format

Submission: **September 30, 2025**

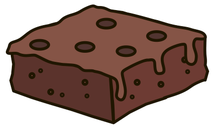
- Topic: brief overview including
  - motivation,
  - different methods, their strengths and weaknesses,
  - discussion of results, and
  - conclusion
- In-class discussion: include relevant ones

Writing Assignment

Report: 7-8 pages (strict), ACM Generic Journal Manuscript Format

Submission: **September 30, 2025**

- Topic: brief overview including
  - motivation,
  - different methods, their strengths and weaknesses,
  - discussion of results, and
  - conclusion
- In-class discussion: include relevant ones
- Future extensions: potential applications of your methods to other topics and/or vice versa



points for ideas!

Guidelines for using Generative AI

For example,

✔ Polish the writing including spelling, grammar, style, translation

✘ Generate new ideas, e.g., the future extension in the report

> “In all cases, students remain responsible for their work. This also applies to the parts of their work that have been created using or influenced by AI.”

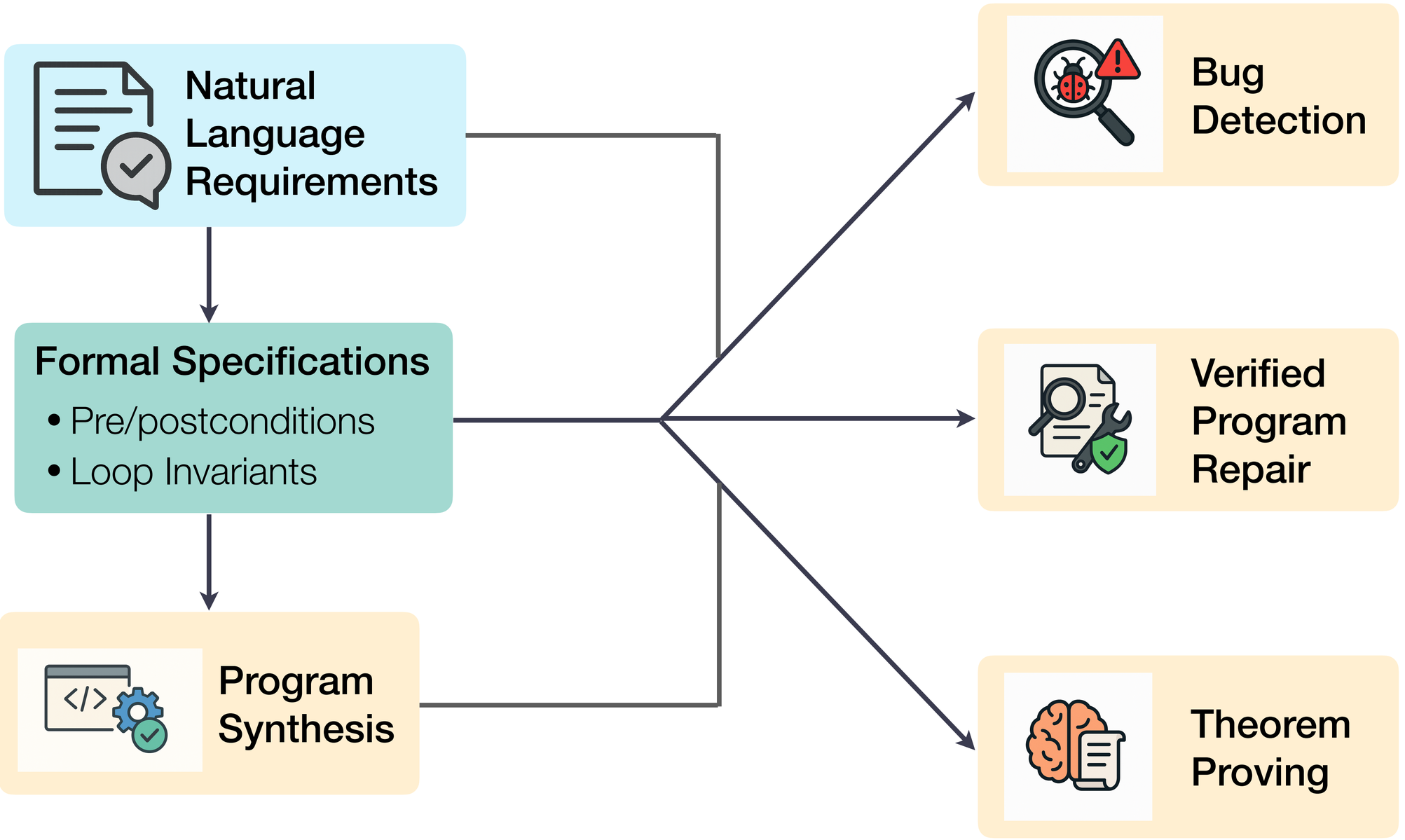
https://www.informatik.kit.edu/downloads/studium/Guidelines_Generative_AI_Informatics.pdf

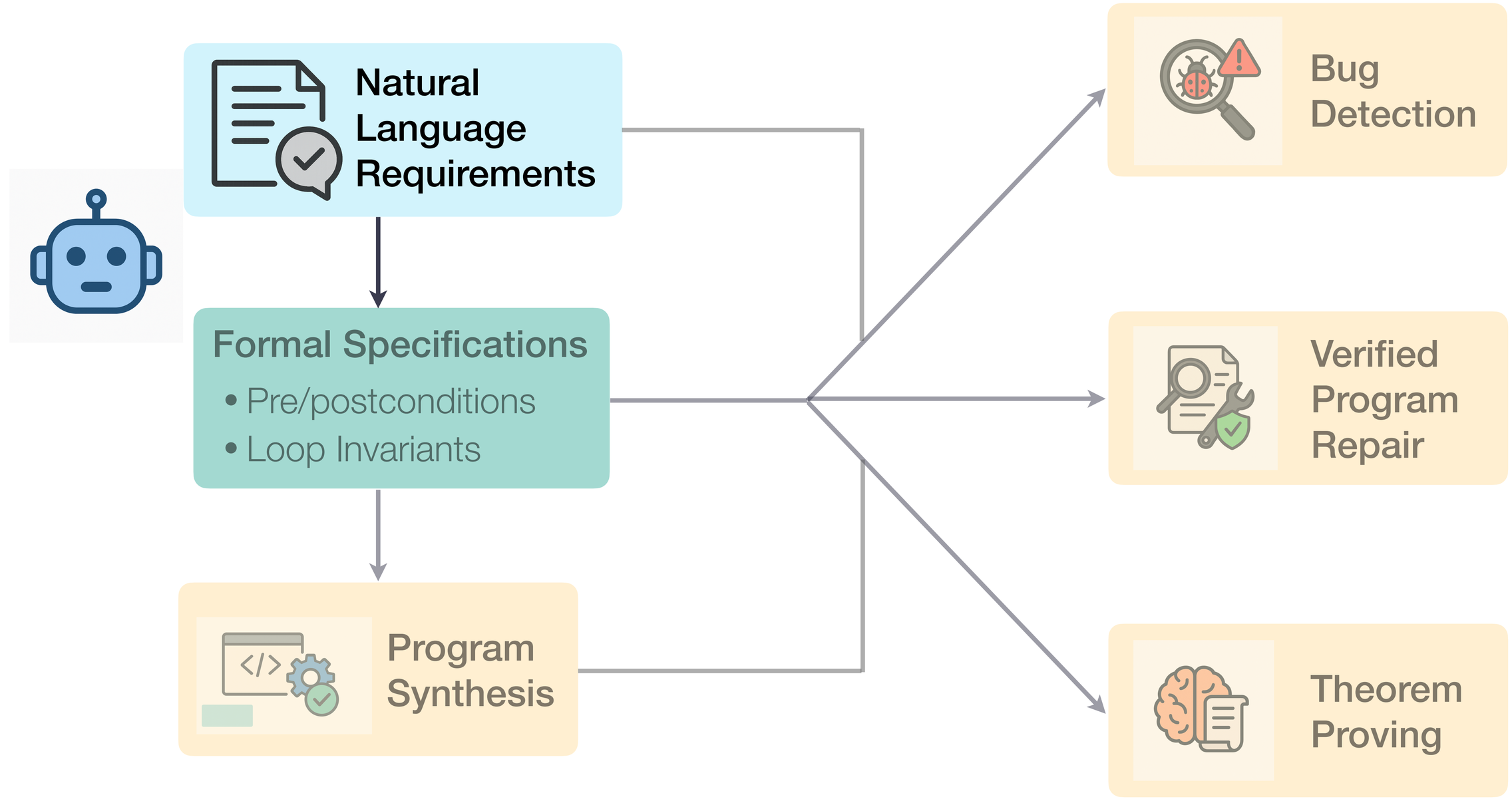
Distribution of Points

- Presentation: 60%
- Report: 30%
- Bonus (in-class discussion): 5%
- Bonus (future extensions in the report): 5%

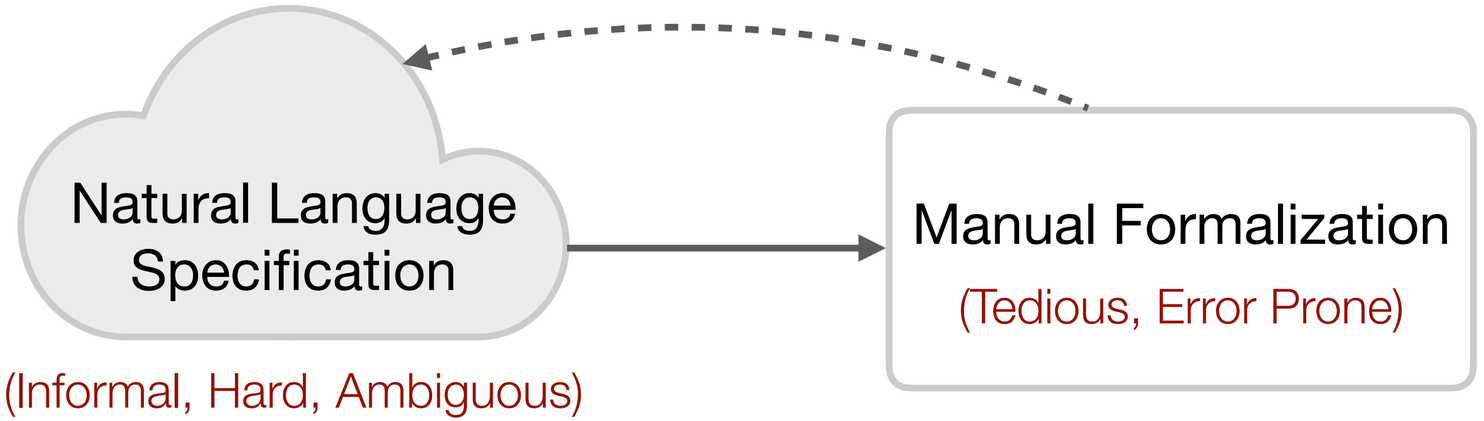
Note: Everybody needs to submit the report to pass!

About the Topics


Natural Language Requirements
Formal Specifications
• Pre/postconditions
• Loop Invariants
Program Synthesis
Bug Detection
Verified Program Repair
Theorem Proving

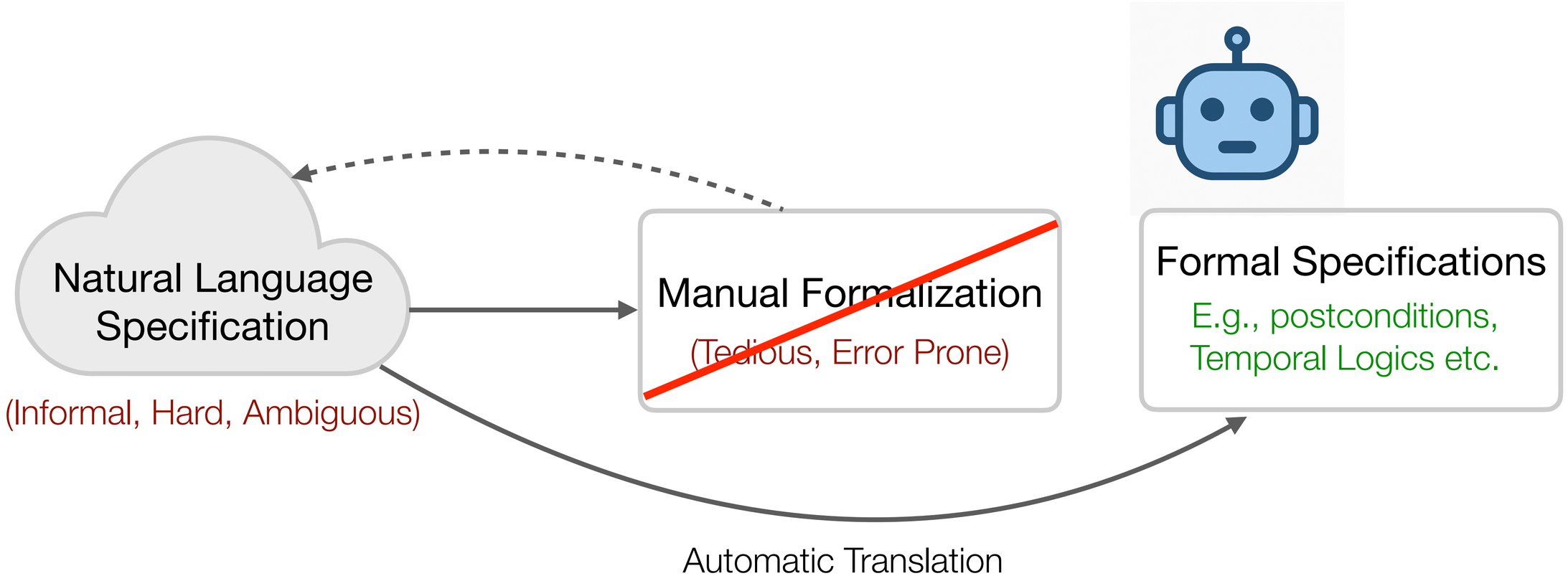

Natural Language Requirements
Formal Specifications
• Pre/postconditions
• Loop Invariants
Program Synthesis
Bug Detection
Verified Program Repair
Theorem Proving

For Natural Language Requirements (Advisor: Debasmita Lohar)

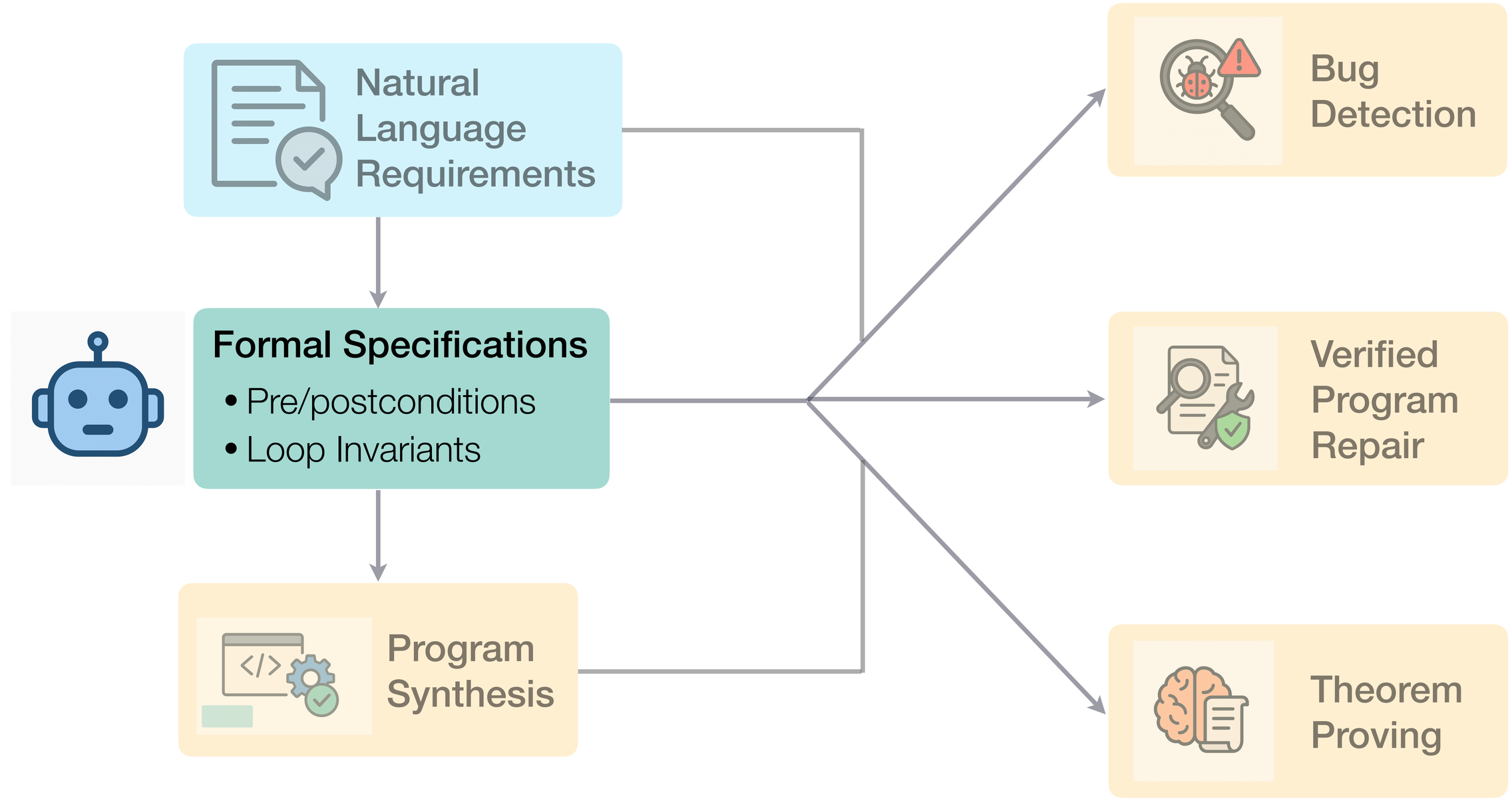


- Can LLMs translate informal specifications into formal?
- Are the specifications are as intended and useful?

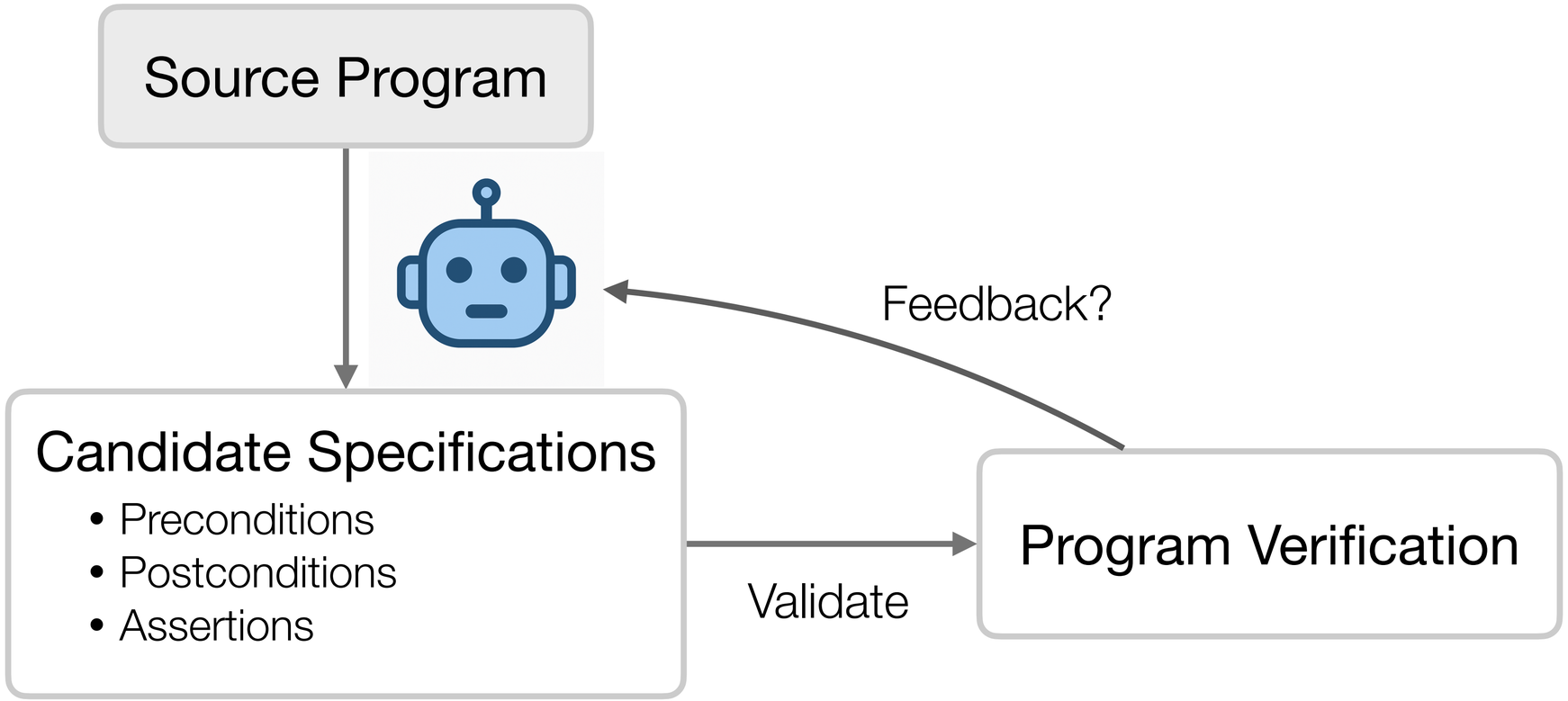
For Natural Language Requirements (Advisor: Debasmita Lohar)



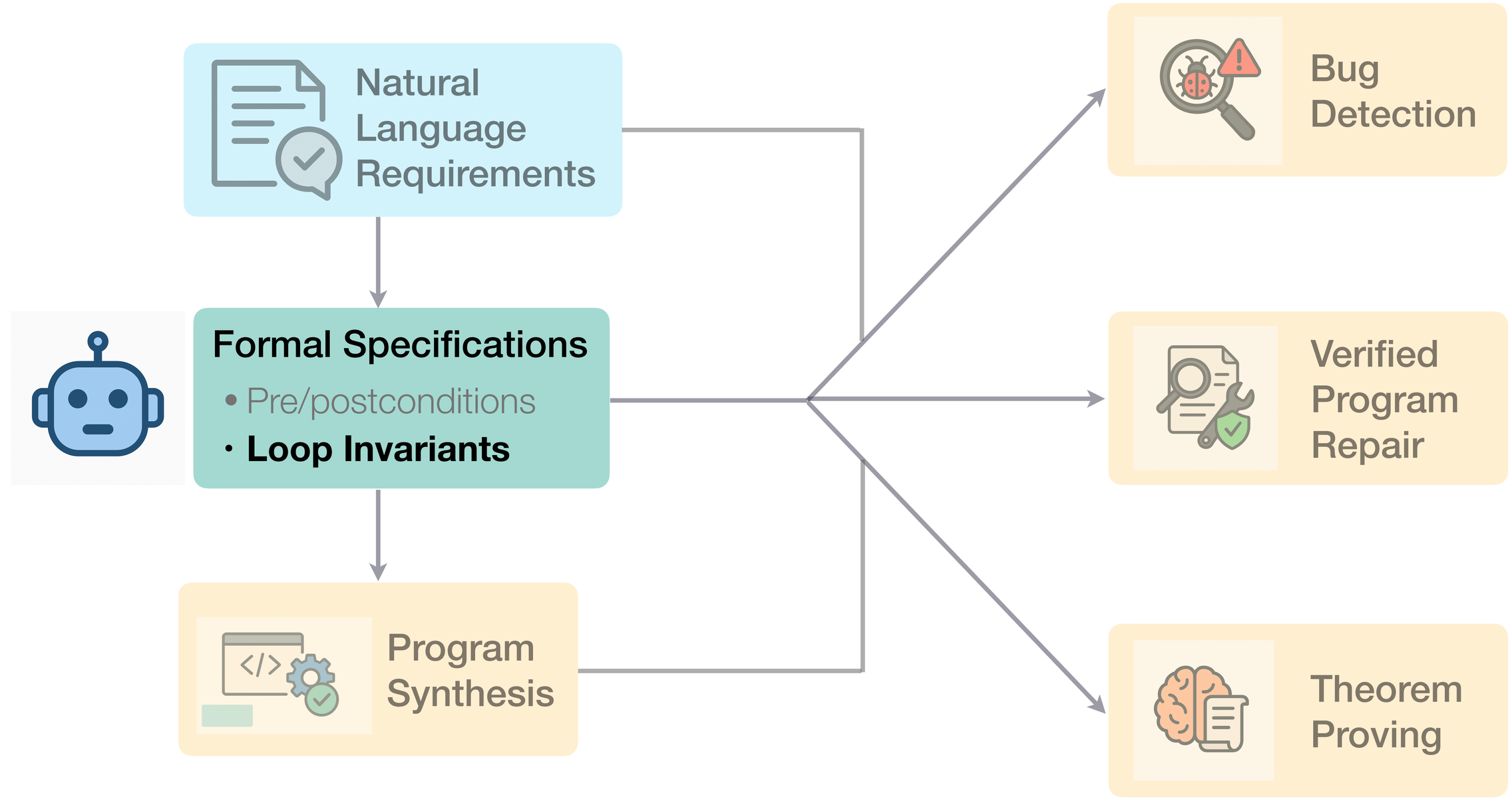
- Can LLMs translate informal specifications into formal?
- Are the specifications are as intended and useful?


Natural Language Requirements
Formal Specifications
• Pre/postconditions
• Loop Invariants
Program Synthesis
Bug Detection
Verified Program Repair
Theorem Proving

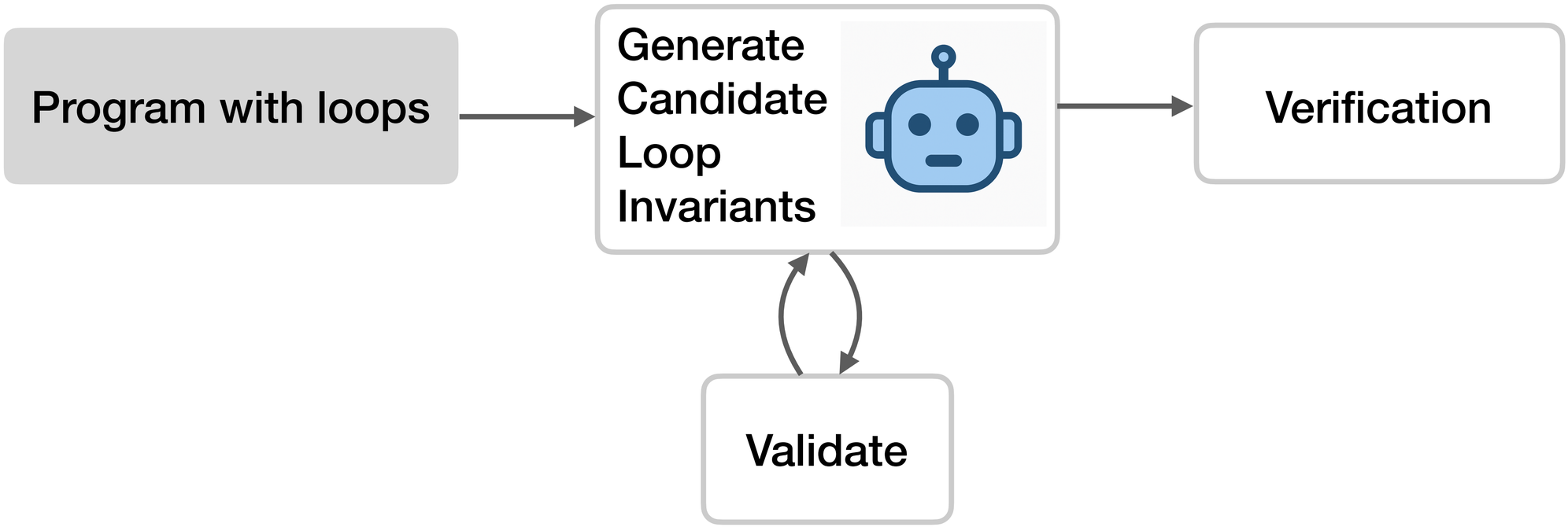
For Program Specifications (Advisor: Samuel Teuber)



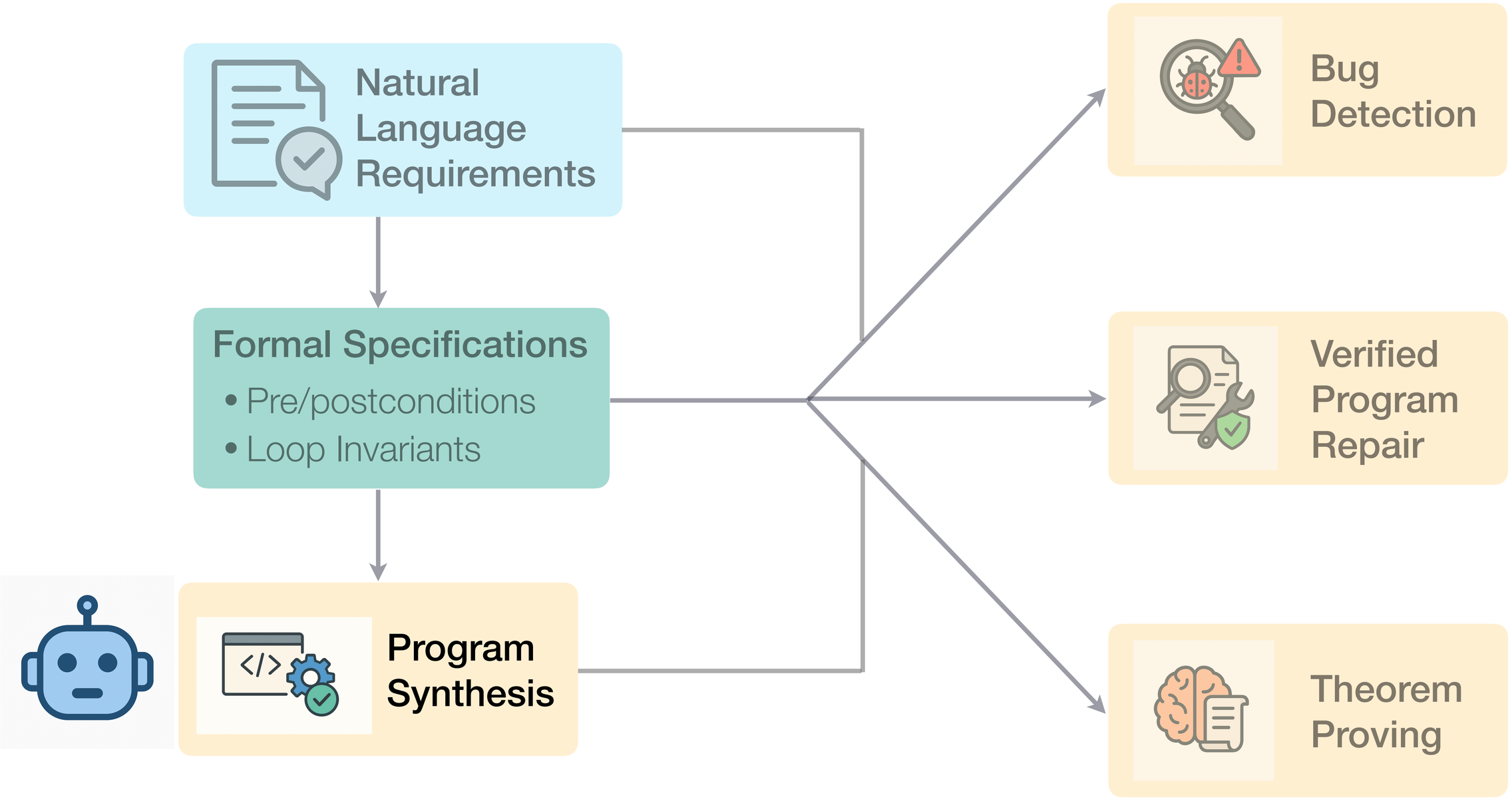
- Can LLMs provide useful intermediate specifications?
- Can LLMs provide further specifications to guide program verification tools?


Natural Language Requirements
Formal Specifications
• Pre/postconditions
• Loop Invariants
Program Synthesis
Bug Detection
Verified Program Repair
Theorem Proving

For Loop Invariants (Advisor: Mattias Ulbrich)



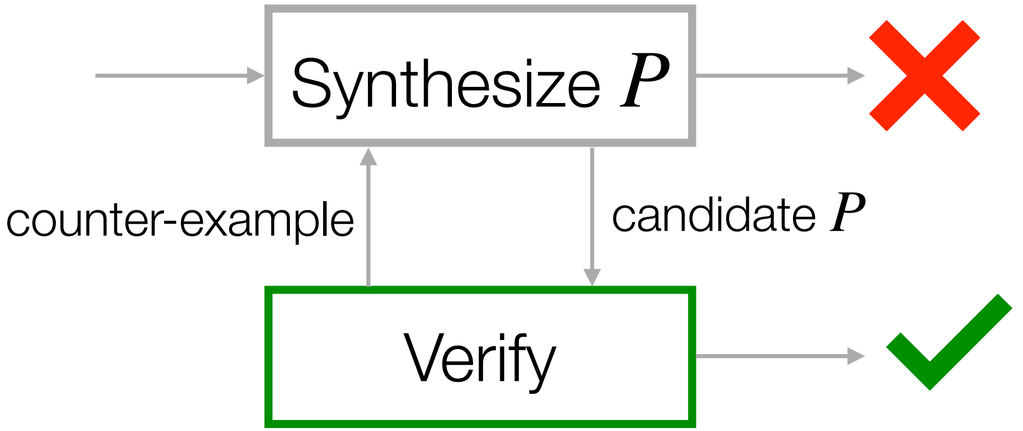
- Can LLMs automatically find useful loop invariants?
- How can these invariants improve the scalability of verification (e.g., BMC)?


Natural Language Requirements
Formal Specifications
• Pre/postconditions
• Loop Invariants
Program Synthesis
Bug Detection
Verified Program Repair
Theorem Proving

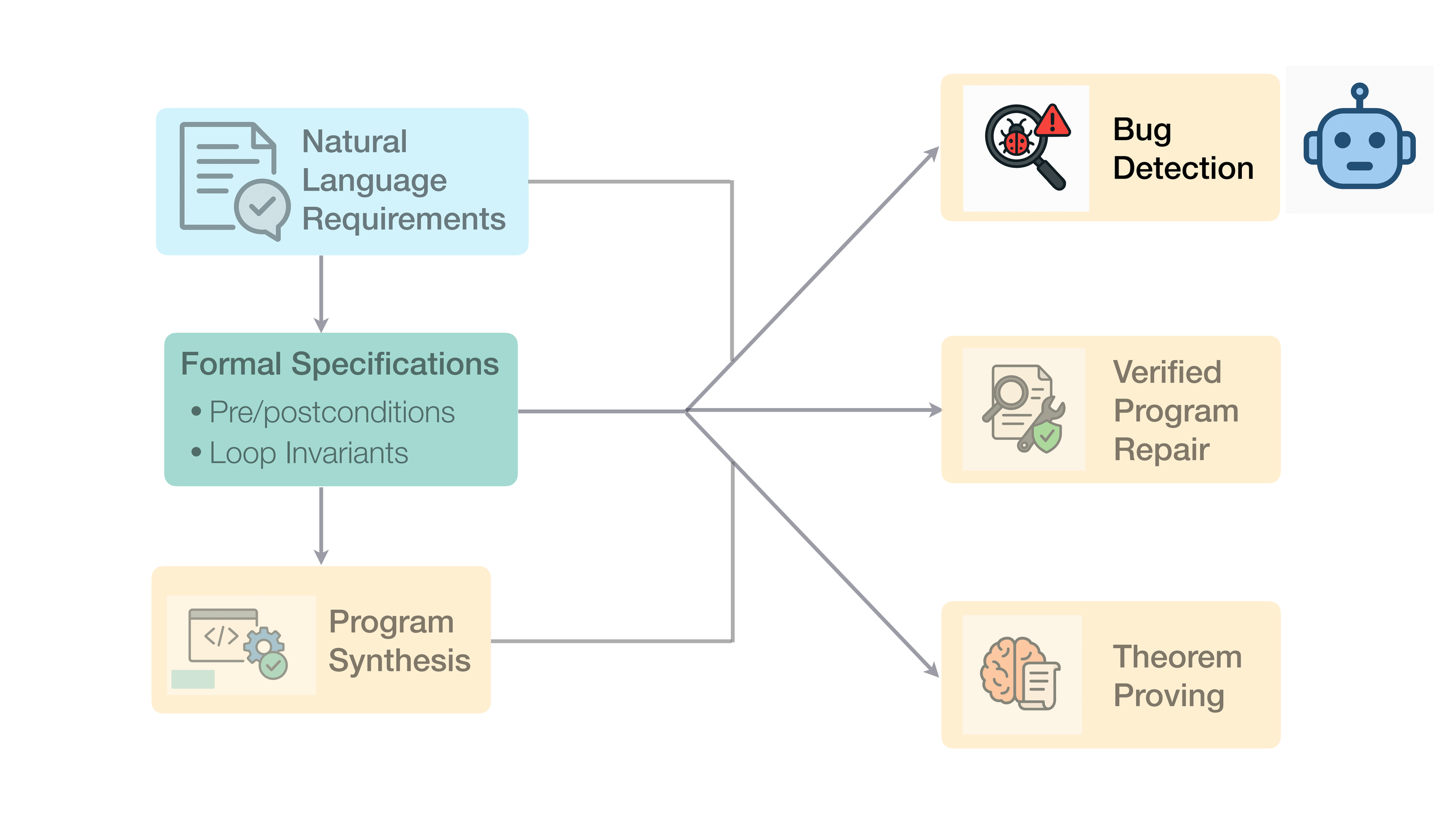
For Program Synthesis (Advisor: Philipp Kern)

$$\exists P \forall x \,.\, \sigma(P, x)$$

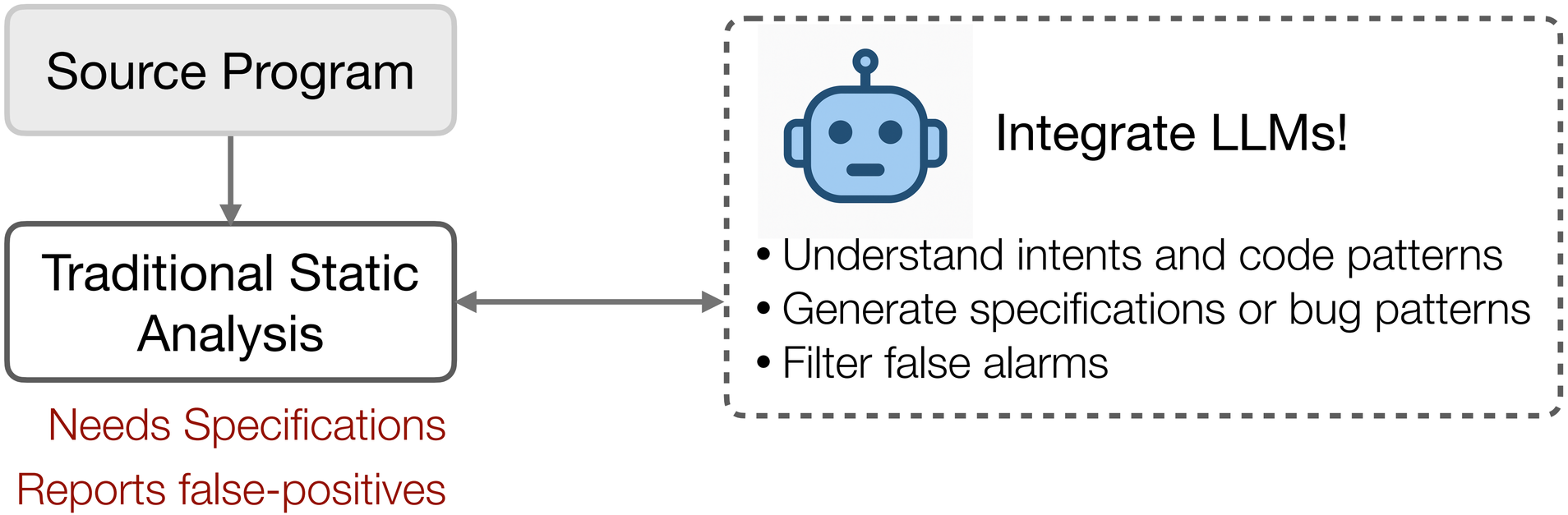
Does there exist a function P such that
for all possible inputs x,
the specification σ will evaluate to true for P and x?



- How can LLMs assist in synthesizing specific code?
- How can we ensure the generated code is correct and consistent?

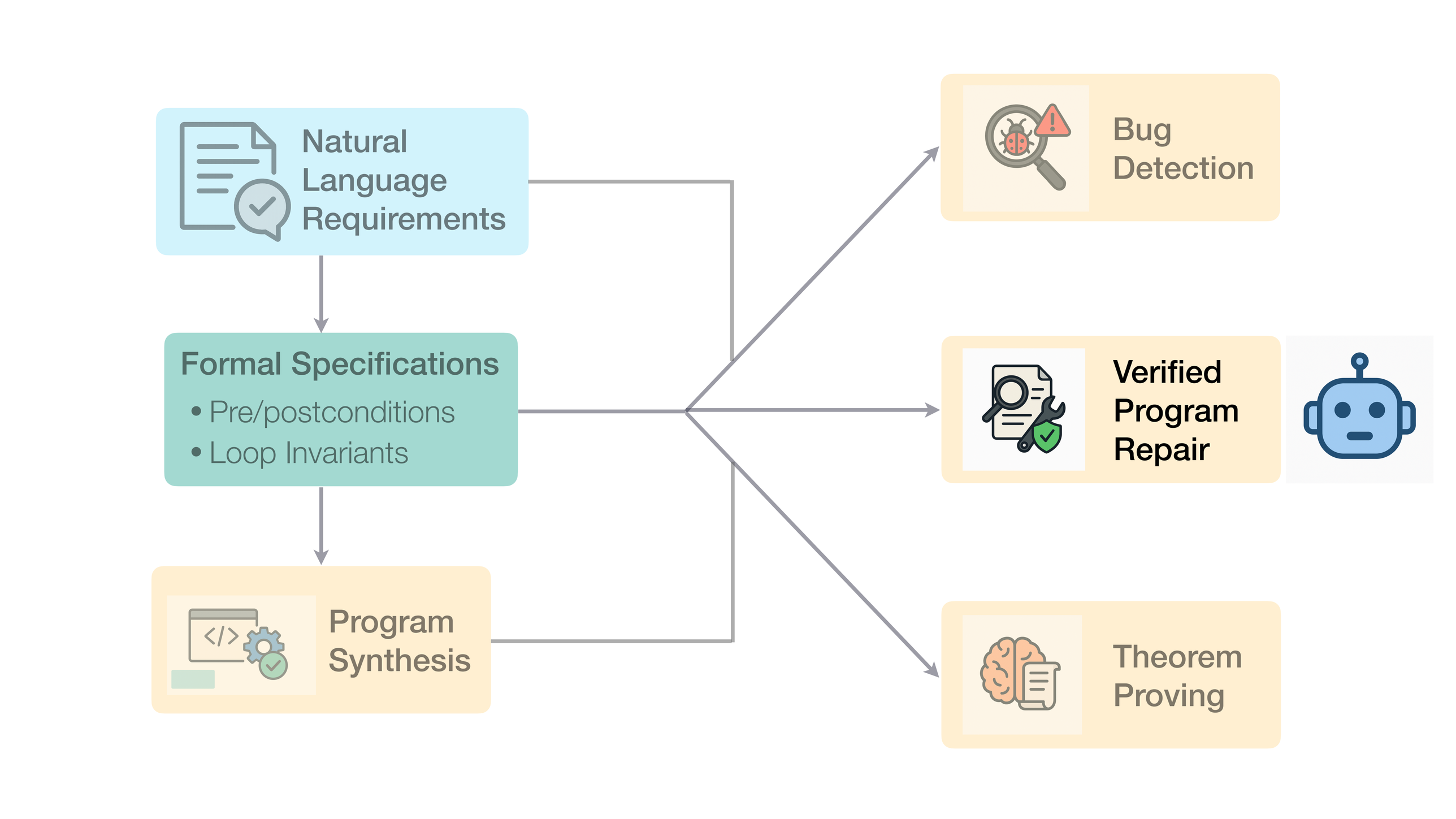

Natural Language Requirements
Formal Specifications
• Pre/postconditions
• Loop Invariants
Program Synthesis
Bug Detection
Verified Program Repair
Theorem Proving

For Bug Detection (Advisor: Jonas Klamroth / Debasmita Lohar)

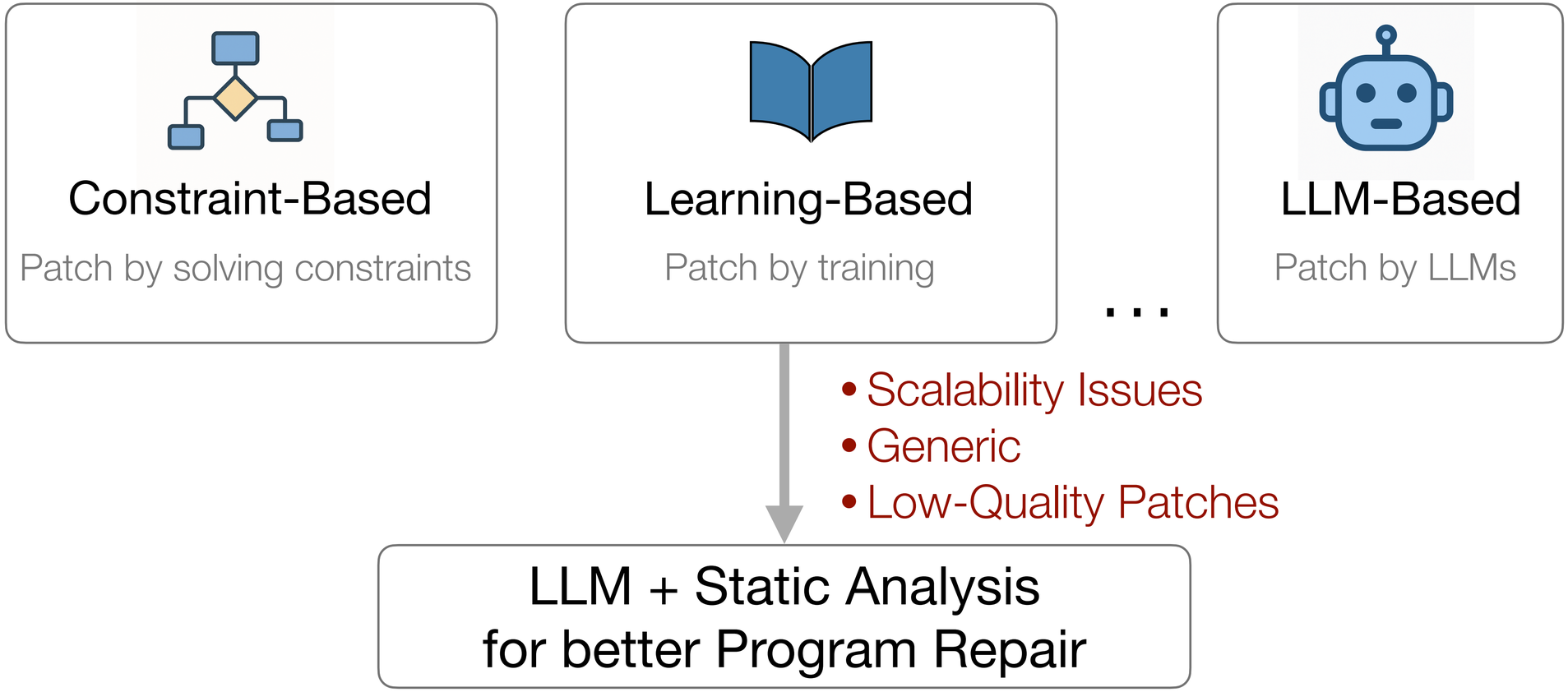


LLM-assisted Static Analysis

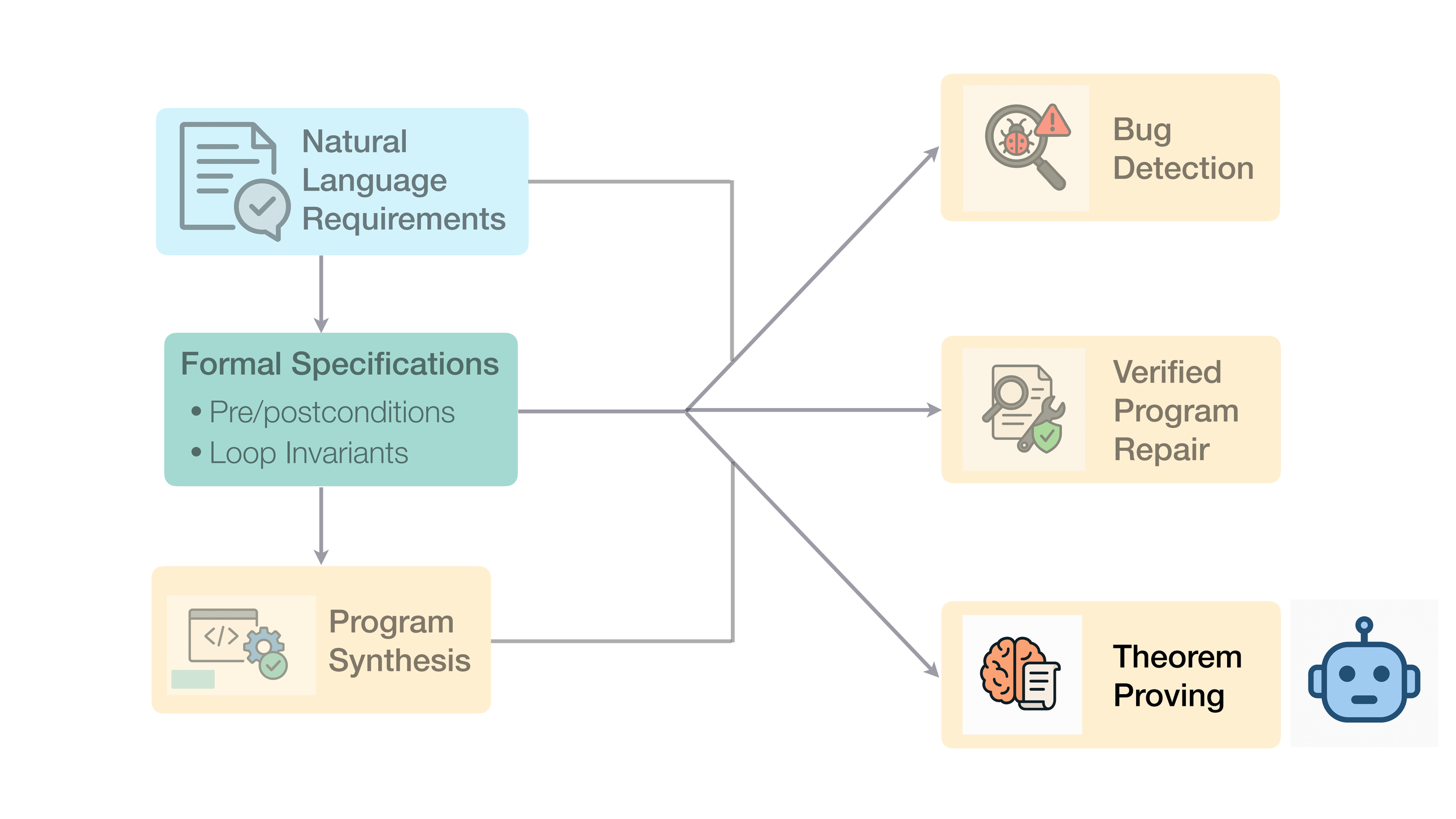
- Can LLMs help static analysis to find bugs without human specifications?
- Are these bugs real or false alarms?


Natural Language Requirements
Formal Specifications
• Pre/postconditions
• Loop Invariants
Program Synthesis
Bug Detection
Verified Program Repair
Theorem Proving

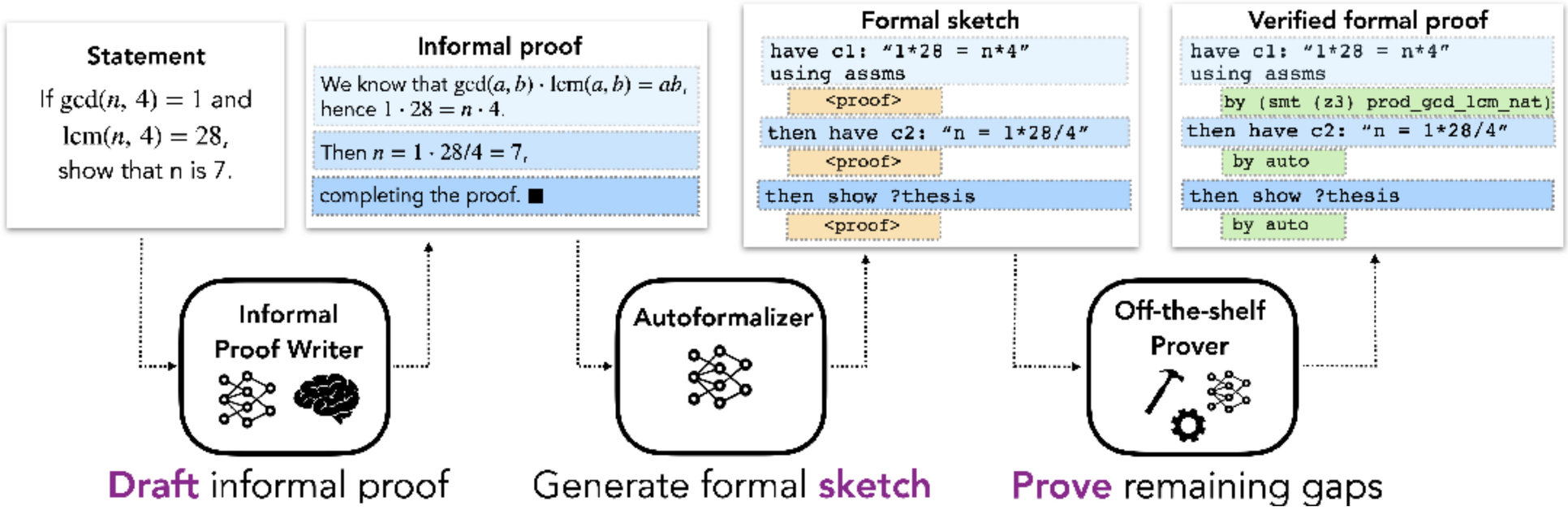
For Verified Program Repair (Advisor: Jonas Schiffl)



- Can LLMs help to generate high quality patches?
- Instead of general, can LLMs provide targeted repair of critical, real-world bugs?


Natural Language Requirements
Formal Specifications
• Pre/postconditions
• Loop Invariants
Program Synthesis
Bug Detection
Verified Program Repair
Theorem Proving

For Theorem Proving (Advisor: Michael Kirsten)



- Can LLMs help in drafting formal theorems and proofs, and finalize existing proofs?
- Can LLMs come up with valid new theorems from existing proof libraries?

Questions?