Synthetic Programming Elicitation for Text-to-Code in Very Low-Resource Programming and Formal Languages

Federico Mora[1] Justin Wong[1] Haley Lepe[2] Sahil Bhatia[1] Karim Elmaaroufi[1]
George Varghese[3] Joseph E. González[1] Elizabeth Polgreen[4] Sanjit A. Seshia[1]

[1]UC Berkeley [2]Stanford University [3]UCLA [4]University of Edinburgh

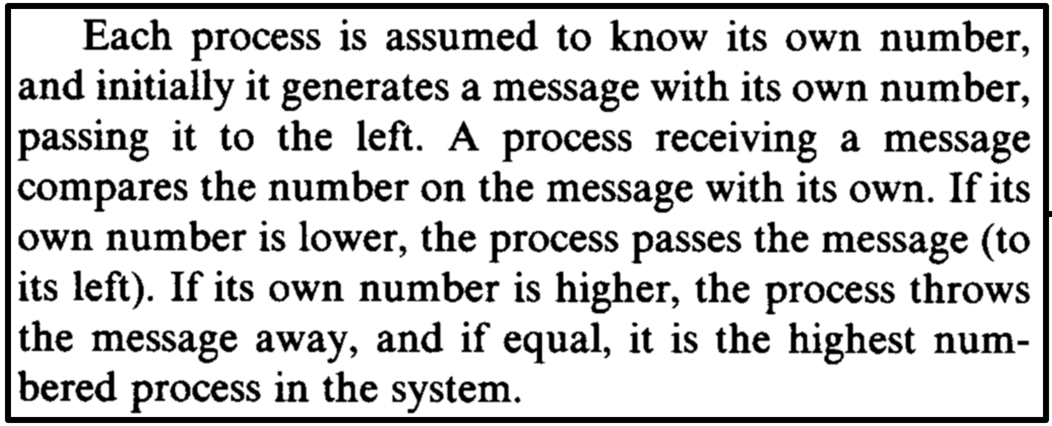
NeurIPS 2024

LLM Code Generation: Just the Tip of the Iceberg

- LLMs are great at generating code in popular languages, like Python.
- Many useful programming languages are not as popular as Python! E.g.,
  - legacy programming languages
  - domain-specific languages (DSLs) for:
    - build systems and tool chains;
    - natural sciences;
    - music and visual art;
    - mathematics;
    - formal verification;
    - and more!
- We want good LLM code generation for all languages!
- Particularly excited about auto-formalization.
  - I.e., text-to-formal-model tools.

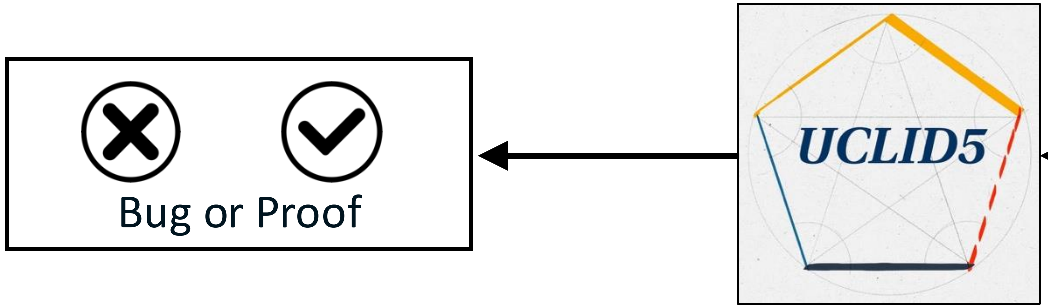
Java
python
JS
php
GO
GNU Make
OCaml
COBOL
SQL
Clojure
LaTeX
GRAPHVIZ
LEAN
THEOREM PROVER
Dafny
mindstorms
NXT 2.0
LMQL
UCLID5

Text-to-Formal-Model Example



Each process is assumed to know its own number, and initially it generates a message with its own number, passing it to the left. A process receiving a message compares the number on the message with its own. If its own number is lower, the process passes the message (to its left). If its own number is higher, the process throws the message away, and if equal, it is the highest numbered process in the system.

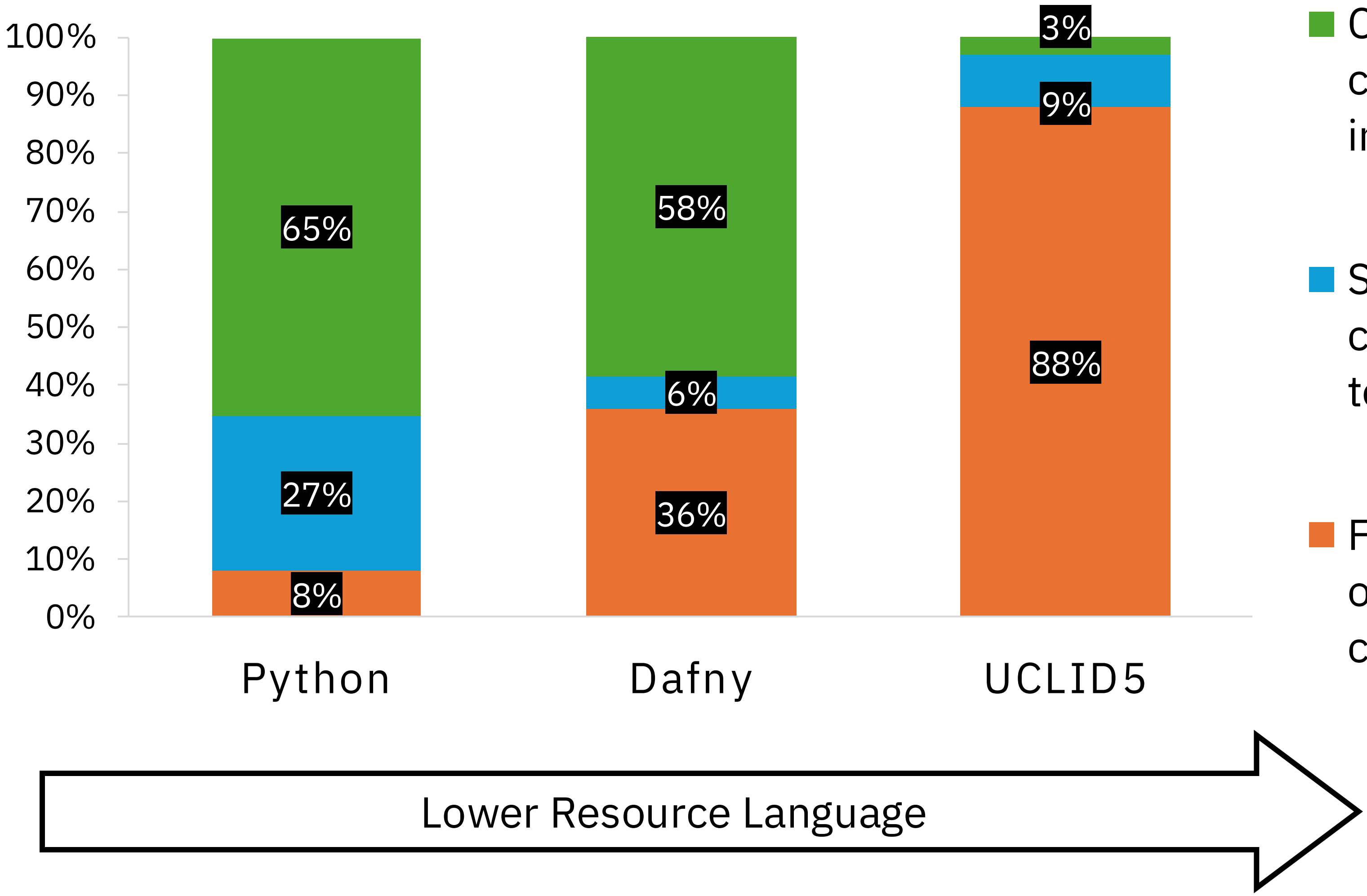
```
module RingModule {
 var id: bv32;
 var m: bv32;
 var stopped: boolean;
 input incoming_m: bv32;
 output output_m: bv32;
 init {
  stopped = false;
 }
 next {
  if (!stopped) {
   if (m == id) {
    stopped' = true;
   } else {
    if (m > id) {
     output_m' = m;
    } else {
     output_m' = id;
    }
   }
  }
  m' = incoming_m;}}
```



Bug or Proof

Chang and Roberts (CACM ‘79); Polgreen et al. (CAV ‘22)

LLMs Perform Poorly on Formal Languages



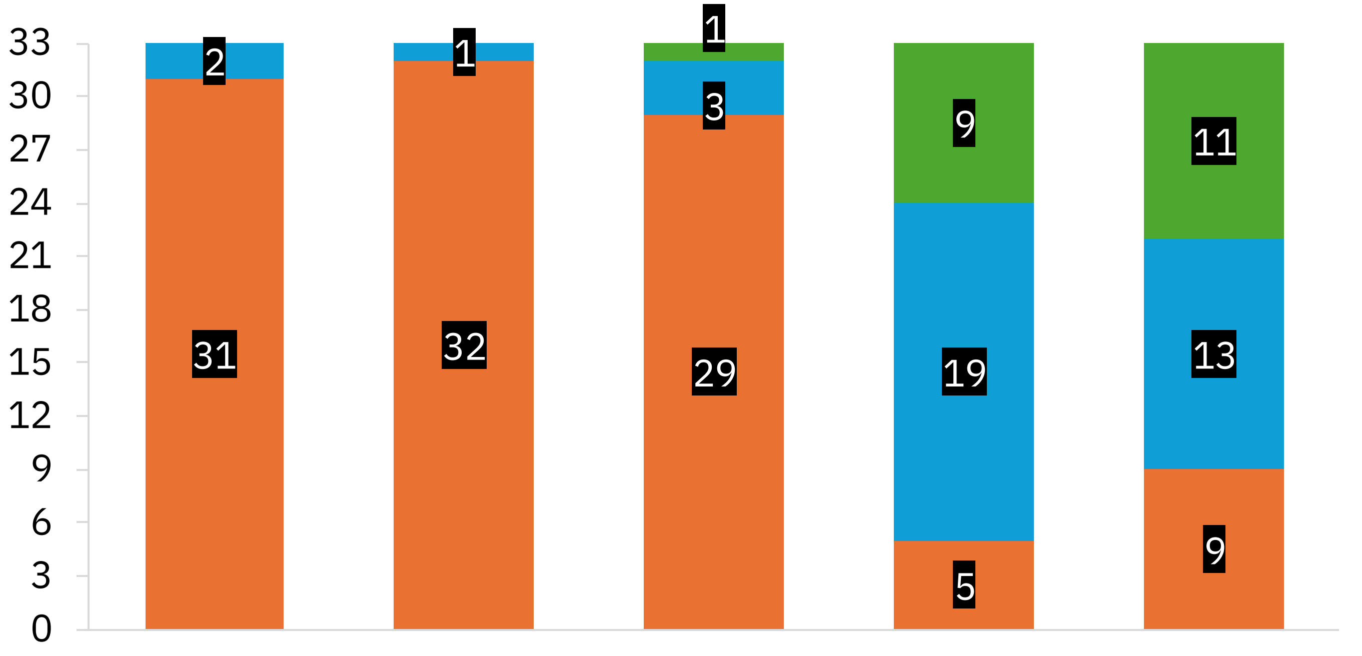
🟩 Correct: passes all compiler checks, tests, and manual inspections.

🟦 Semantic Errors: passes compiler checks but fails tests or manual inspection.

🟧 Failures: fails compiler checks or otherwise fails to generate code (e.g., timeout).

Duo et al. (Arxiv ’24); Miso et al. (FSE ’24); Polgreen et al. (CAV ‘22); Leino (LPAR ‘10); github.com/uclid-org/uclid; github.com/dafny-lang/dafny

Text-to-UCLID5 Evaluation



🟩 Correct: passes all compiler checks, tests, and manual inspections.

🟦 Semantic Errors: passes compiler checks but fails tests or manual inspection.

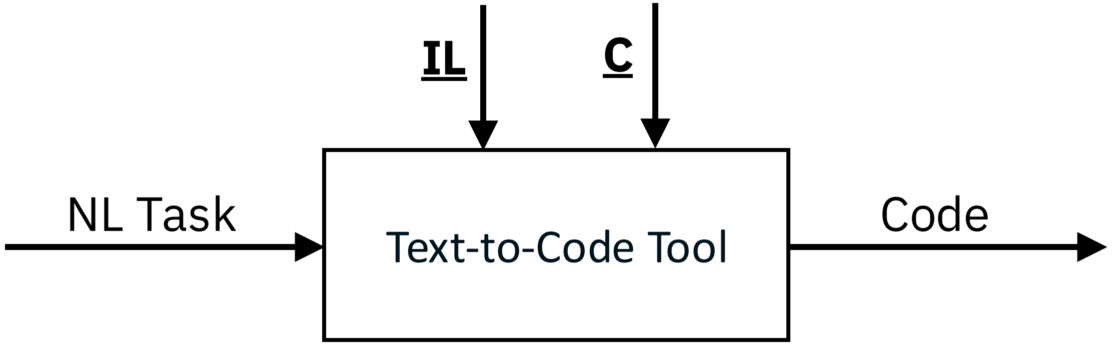
🟧 Failures: fails compiler checks or otherwise fails to generate code (e.g., timeout).

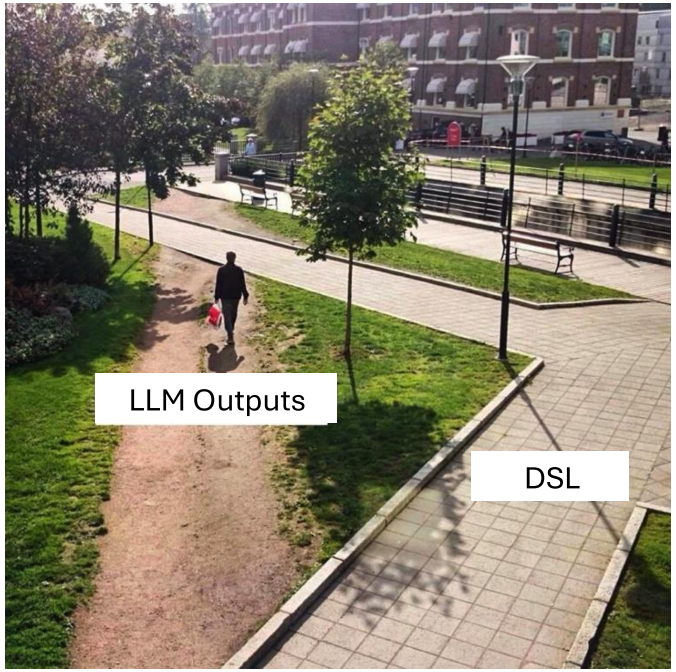
Lee and Seshia (‘16); Huth and Ryan (‘04); Baier and Katoen (’08)

Insight #1: Design an Intermediate Language for the LLM!

- Existing work: natural programming elicitation.
  1. Understand what programmers find "natural."
  2. Design a programming language or tool for that.

- Our work: synthetic programming elicitation.
  1. Understand what your LLM finds "natural" in your target domain.
  2. Design an intermediate language (**IL**) that matches that.
  3. Write a compiler (**C**) from the IL to your target language.

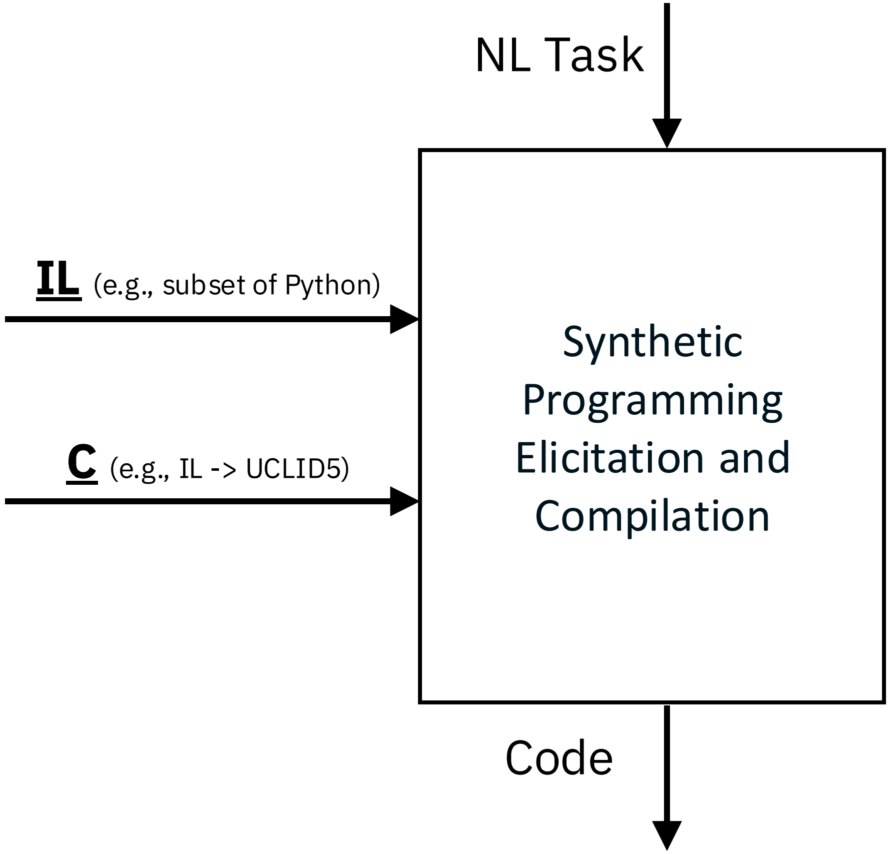




> "The premise of our research project is that programmers will have an easier job if their programming tasks are made more natural."
>
> – Myers, Pane, and Ko

Myers, Pane, and Ko (CACM '04)

Simplified Hypothetical Example



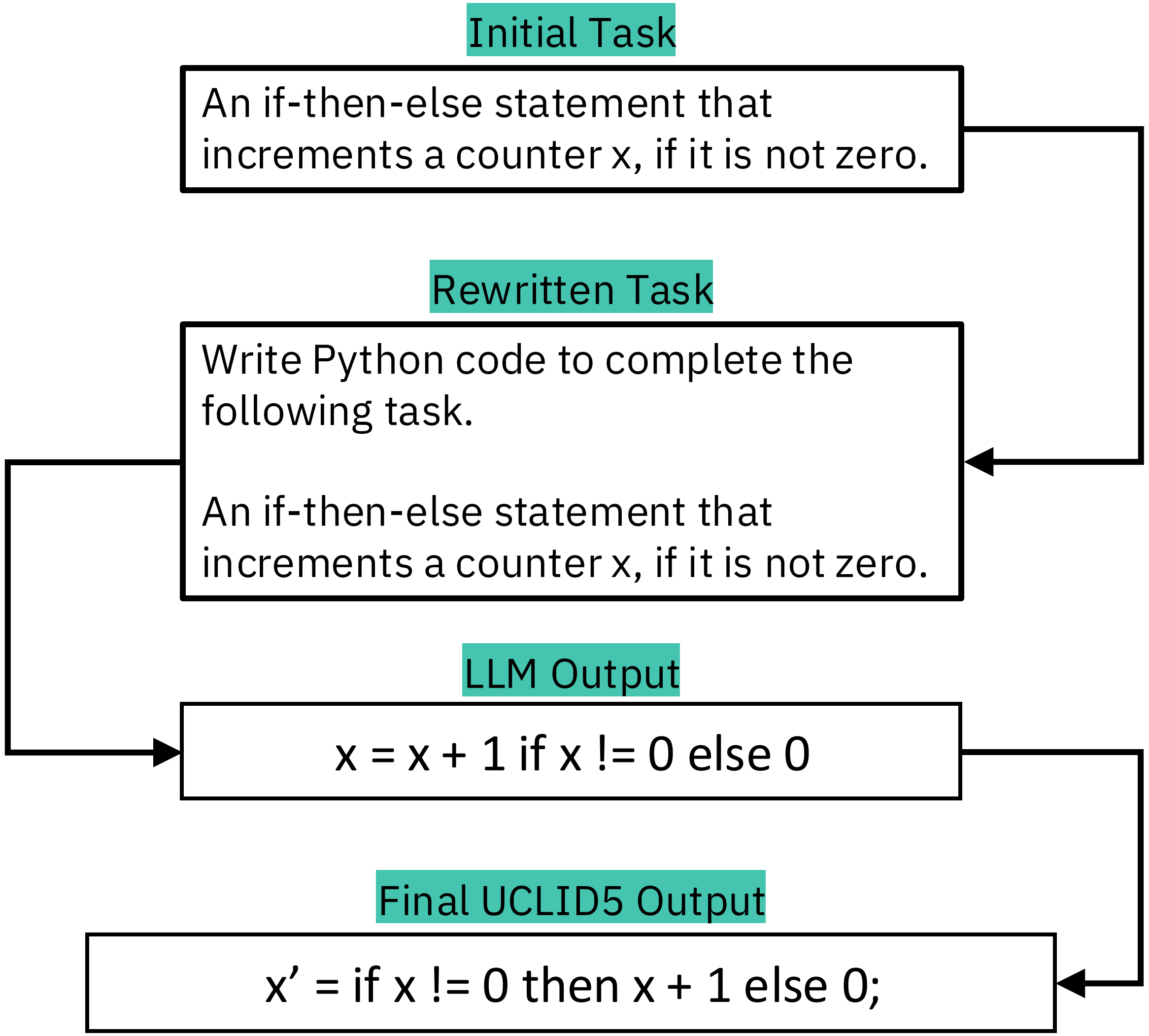
NL Task

IL (e.g., subset of Python)

C (e.g., IL -> UCLID5)

Synthetic Programming Elicitation and Compilation

Code



Initial Task

An if-then-else statement that increments a counter x, if it is not zero.

Rewritten Task

Write Python code to complete the following task.

An if-then-else statement that increments a counter x, if it is not zero.

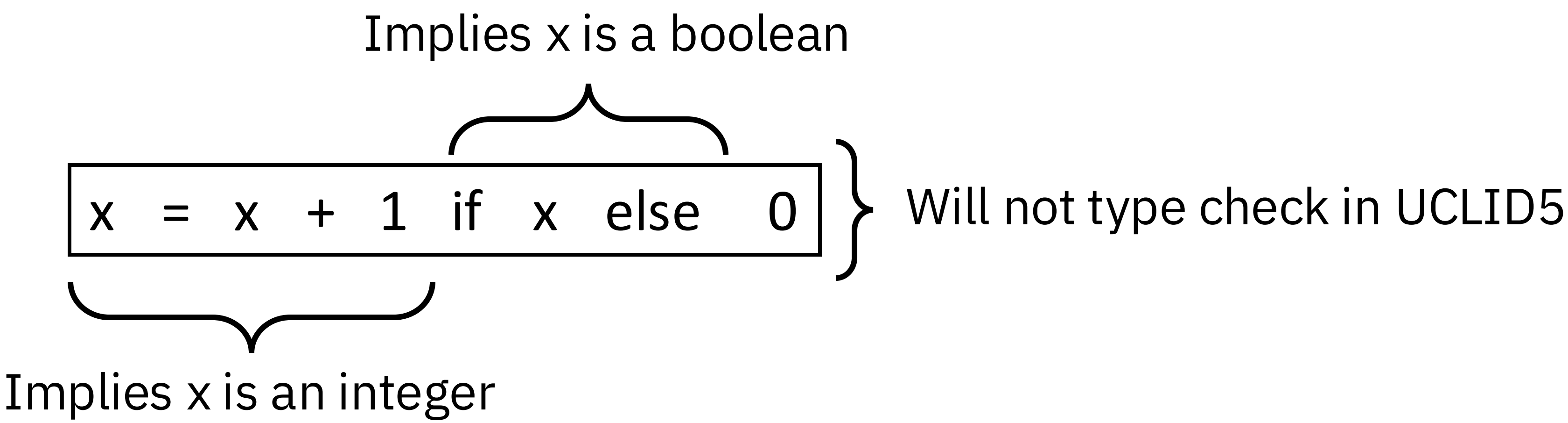
LLM Output

```
x = x + 1 if x != 0 else 0
```

Final UCLID5 Output

```
x’ = if x != 0 then x + 1 else 0;
```

Intermediate and Target Languages Differ



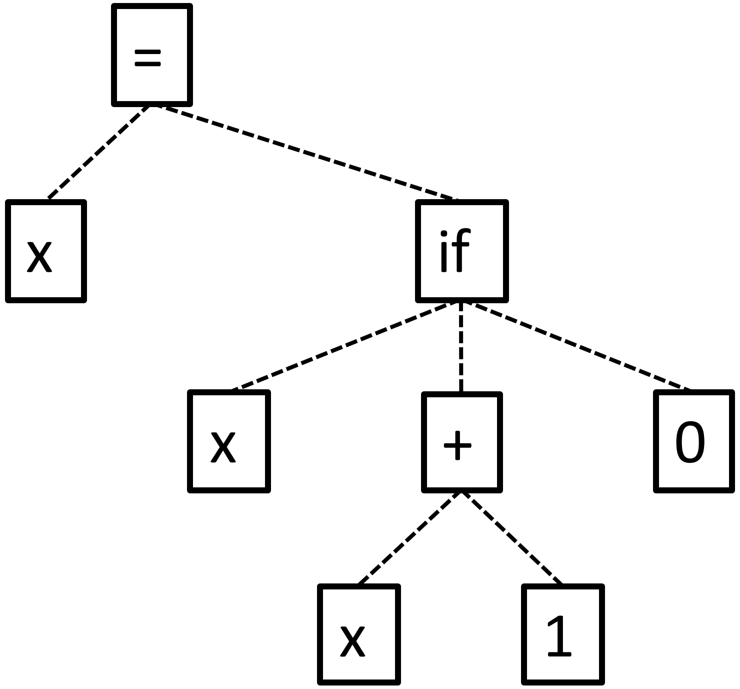
Implies x is a boolean

```
x = x + 1 if x else 0
```

Will not type check in UCLID5

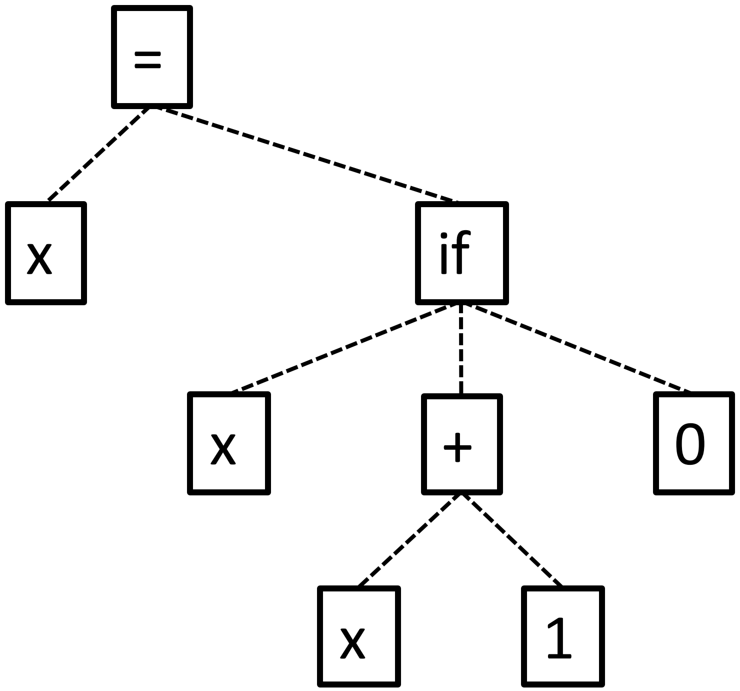
Implies x is an integer

Insight #2: Find Minimal Error Sources and Repair

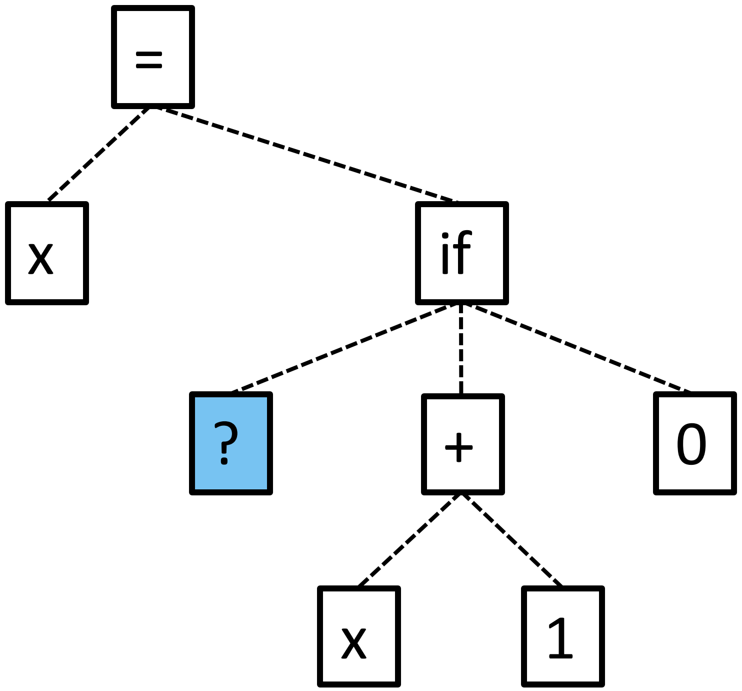


Parse Program

Insight #2: Find Minimal Error Sources and Repair



Parse Program



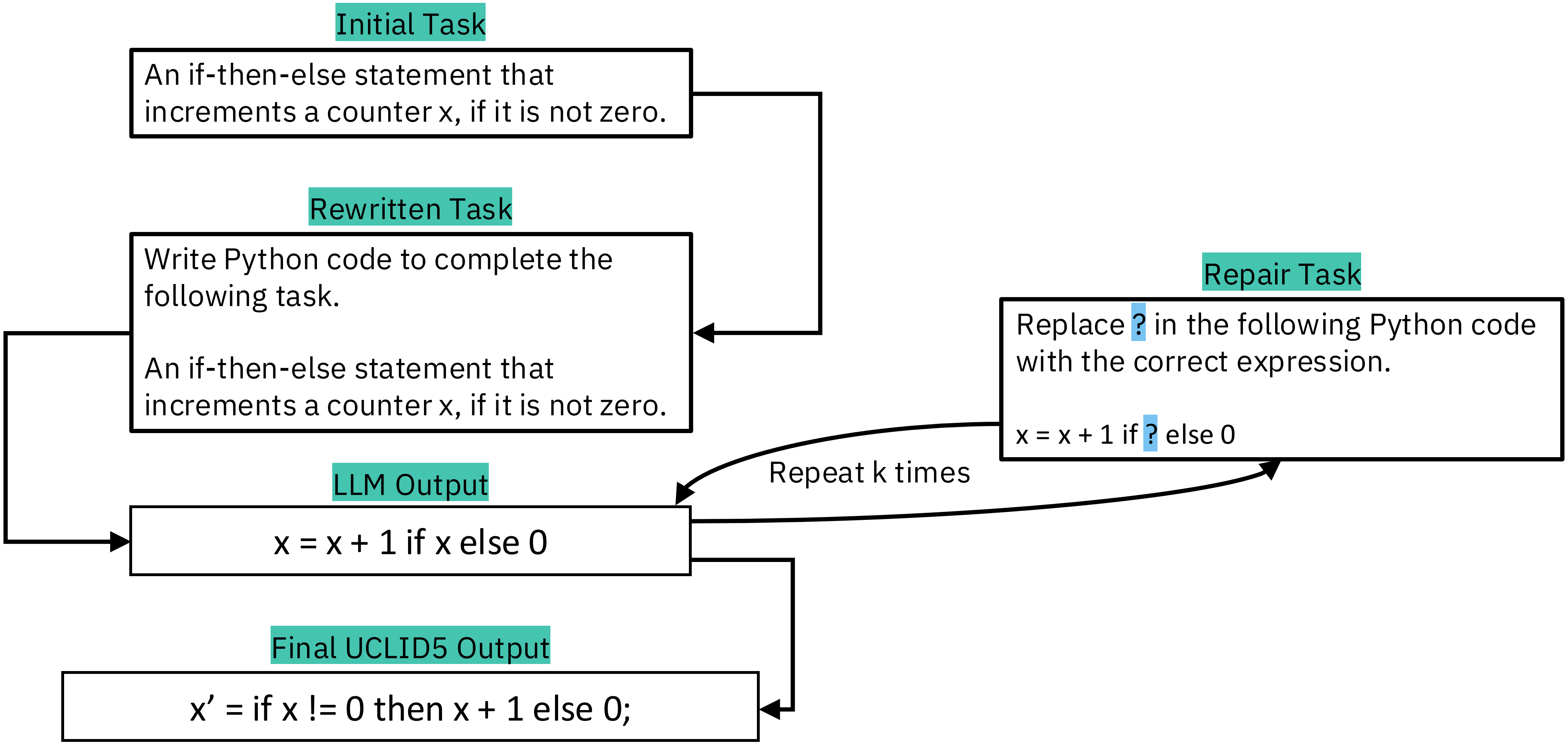
Cut Program with MAX-SMT

Hi LLM, please replace ? in the following code with the correct expression.

```
x = x + 1 if ? else 0
```

Repair Program

Simplified Hypothetical Example, Revisited



Initial Task

An if-then-else statement that increments a counter x, if it is not zero.

Rewritten Task

Write Python code to complete the following task.

An if-then-else statement that increments a counter x, if it is not zero.

Repair Task

Replace ? in the following Python code with the correct expression.

x = x + 1 if ? else 0

Repeat k times

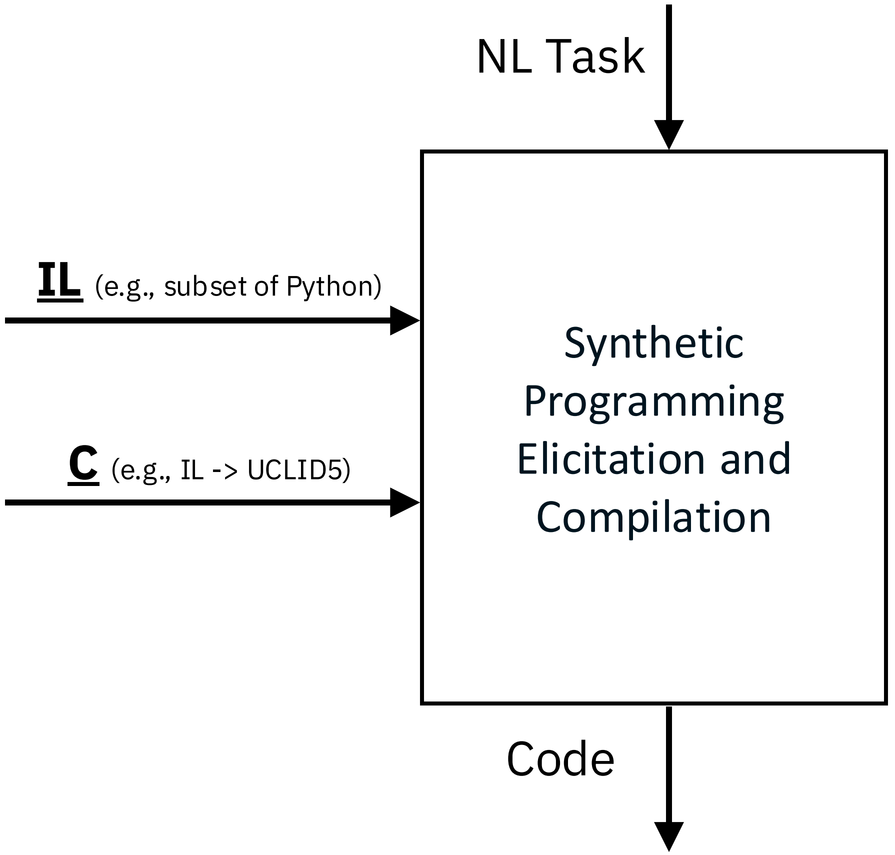
LLM Output

x = x + 1 if x else 0

Final UCLID5 Output

x’ = if x != 0 then x + 1 else 0;

Summary: Language Design and Symbolic Techniques Can Help LLMs Write Code!



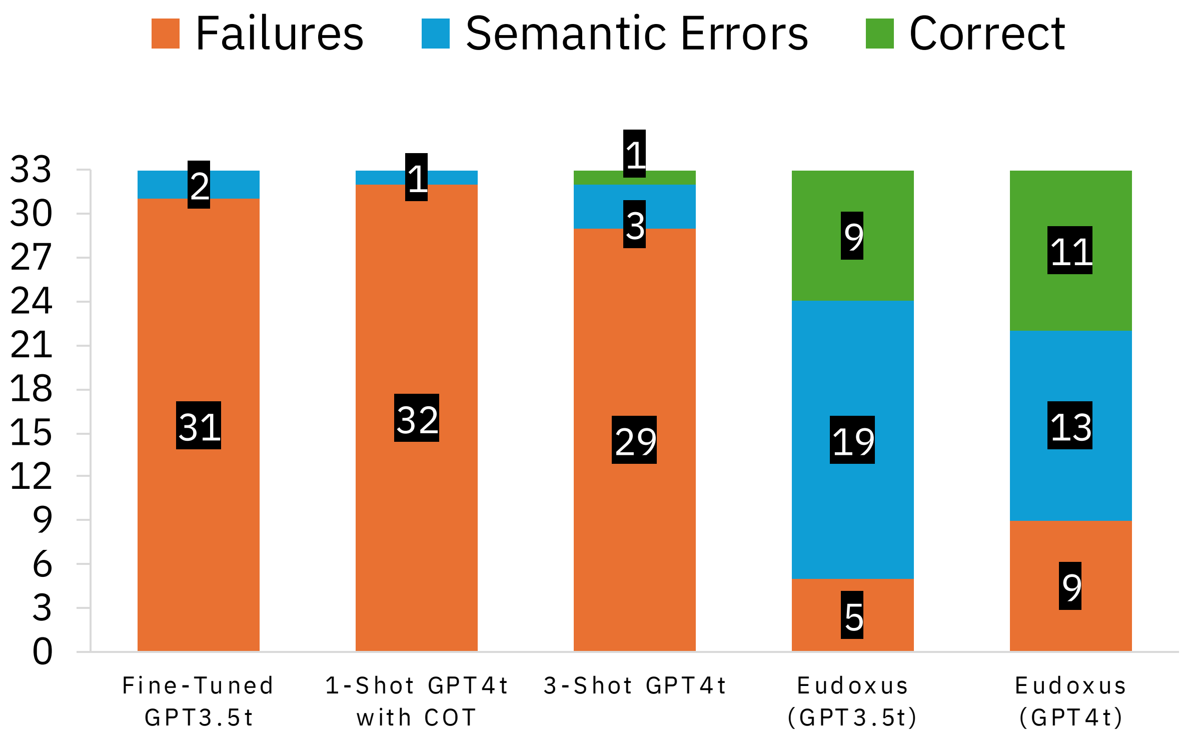
NL Task

IL (e.g., subset of Python)

C (e.g., IL -> UCLID5)

Synthetic Programming Elicitation and Compilation

Code


Failures
Semantic Errors
Correct
Fine-Tuned GPT3.5t
1-Shot GPT4t with COT
3-Shot GPT4t
Eudoxus (GPT3.5t)
Eudoxus (GPT4t)