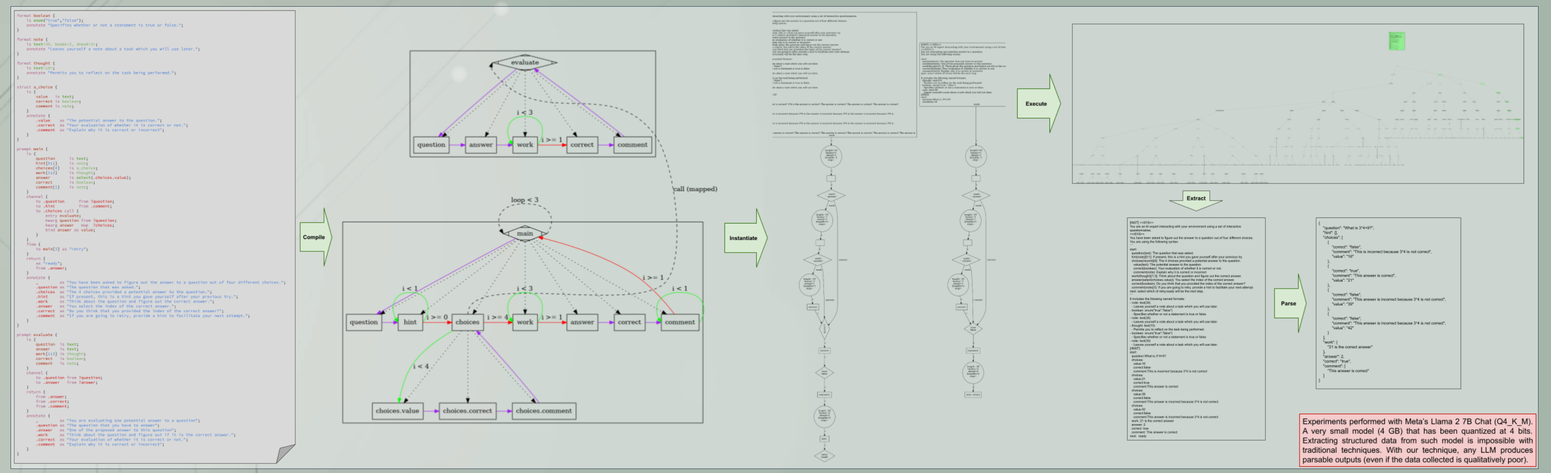
Cognitive Programming Model for Reliable and Inspectable Automated Decision Making using Large Language Models

Tristan Vanderbruggen & Matt Sottile, Lawrence Livermore National Laboratory
https://github.com/llnl/autocog

- LLMs show limited abilities to reason and use tools
  - LLMs can hallucinate instructions and forget the original request
  - LLM reasoning can get poisoned by hallucination (aka errant thoughts)
- Complex instructions are used to make LLMs follow commands
  - Relies on JSON and "function call" to obtain actionable data from the model
- We encode instructions into the problem solved by the LLM instead
  - Defining complex syntax using automaton makes parsing LLM's outputs reliable

- Automaton & Cognition (AutoCog):
  - Framework based on automata theory to construct cognitive applications
  - Defines an execution model for LLM: Finite Thought Automata (FTA)
- Structured Thoughts:
  - Programming Model for Software-defined LLM prompting
  - Compile each prompt to FTA which is executed by any LLM
  - Structured language with reusable components to create libraries

- Software-defined prompting
  - Declares *prompts* as structured documents
  - Declares types for faster coding
  - Defines control-flow, data-flow, and calls
    - Prompts are analogous to traditional basic-blocks
  - Natural Language Annotations provide semantic
- Structured Thought Automata (STA)
  - How to traverse each prompt's data-structure
  - Purple edge always taken
  - Green/Red edges are branch
    - Condition based on visit
- Finite Thought Automaton (FTA)
  - DAG of actions executed by the LLM
    - Box: known text
    - Diamond: choice between a set of values
    - Ellipsis: completion with beam search variant
  - Instantiated for specific input data
- Finite Token Tree (FTT)
  - Probabilistic exploration of the FTA
    - Nodes correspond to FTA's actions
    - Nodes have list of tokens and probabilities
  - Path of max probability is the "solution"



Experiments performed with Meta's Llama 2 7B Chat (Q4_K_M).
A very small model (4 GB) that has been quantized at 4 bits.
Extracting structured data from such model is impossible with traditional techniques. With our technique, any LLM produces parsable outputs (even if the data collected is qualitatively poor).

Current Work: Programming Model

- Seamlessly calling python from ST
  - Dataflow helpers: enumerate, zip, ...
  - Interact with environment
  - Generally using ***tools***
- Searching and Pruning
  - Prevent bloated FTT
  - Preserve result quality
- High Performance execution engine
  - Currently too slow for production
  - Planning C++ within Llama.cpp

Future Work: Specialized LLMs

- Finetuning LLM for ST
  - Reinforce best branch of FTT
- Bootstrapping LLM using ST
  - Using "spiral curriculum" (elementary school)
- ST specific Tokenization
  - Protocol above Natural Language

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-POST-863621

THE TWENTY-FOURTH ANNUAL
HIGH CONFIDENCE SOFTWARE AND SYSTEMS CONFERENCE
MAY 6 - 8, 2024 | http://cps-hcss.org