LangGraph - Task 9

(Auditors - Sameera Sudarshan Kashyap and Pavithra Selvaraj) | GitHub

LangGraph helps in building production ready orchestration of stateful agents. It provides fault tolerance allowing any agent to continue performing a task and flexibility of humans in the loop to seamlessly incorporate human oversight by inspecting and modifying agent state at any point during execution.

Migration

To migrate our program to LangGraph, we decide the architecture of the program, in our code we have 5 steps

- Classifier - Classifies if the natural language question is a Logic or Constraint problem.
- Solvers
    - Logic Solver - Create the prompt to query the LLM with relevant problem statements and context obtained from the RAG database.
    - Constraint Solver - If the obtain question is a Constraint problem, then the constraint solver method is invoked
- Refinement - If the result obtained is uncertain, then self refinement, reprompts the LLM to obtain a better answer until the LLM can provide a deterministic boolean.
- Result Interpreter - Based on the boolean answer, the interpreter provides a solution back in natural language.

To create a Graph from the existing LangChain implementation, we install langraph library

```shell
pip install langchain langgraph
```

We then declare a Graph with a State class according to the input fields required by our program

```python
class State(TypedDict):
    question: str
    classifier: str
```

```
    context: str
    csp_response: str
    logic_response: str
    result_interpreted: str
```

We then instantiate a `StateGraph` and add the required nodes and set the entry point.

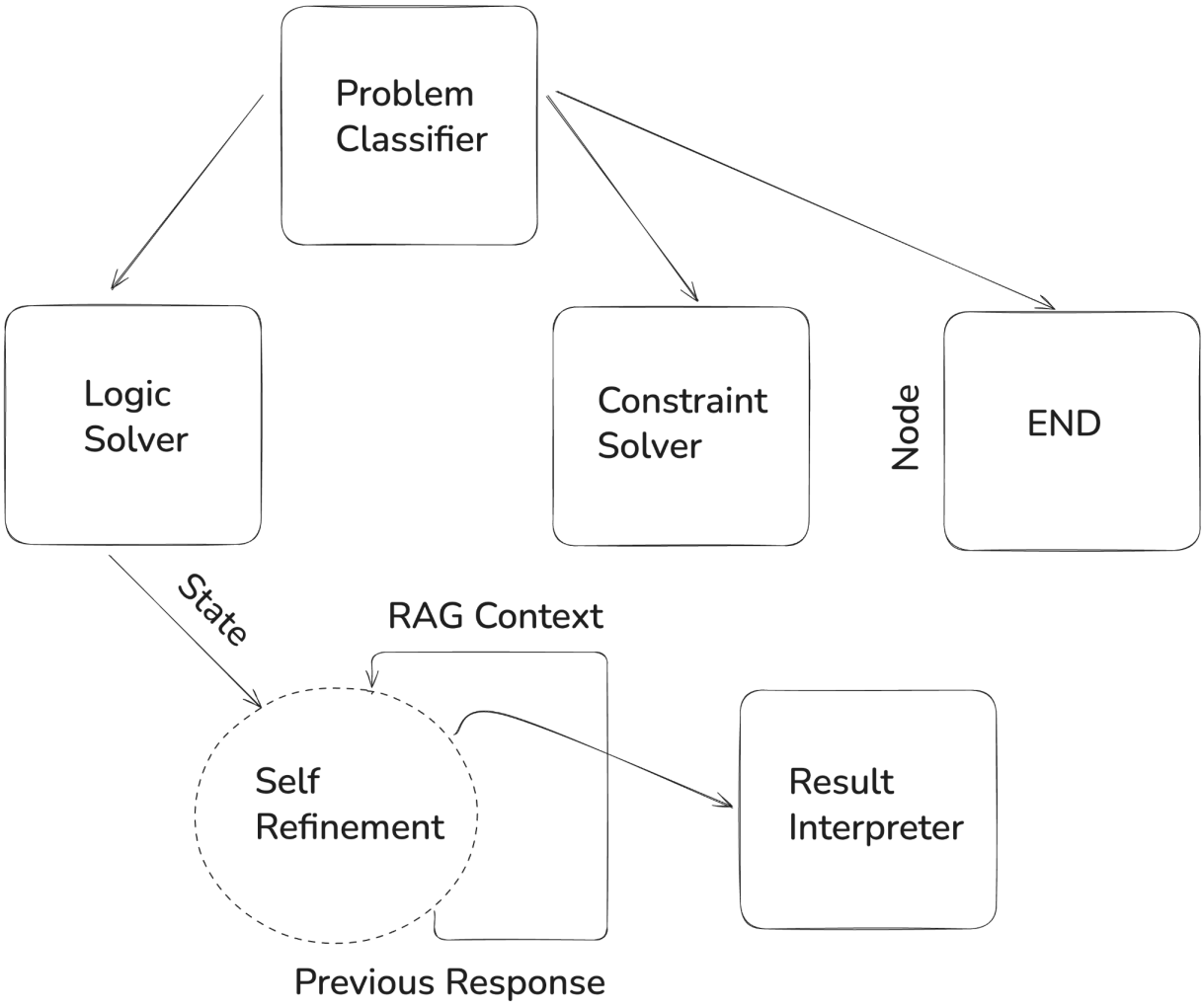
```Python
builder = StateGraph(State)

builder.add_node("problem_classifier", problem_classifier)
builder.add_node("constraint_solver", constraint_solver)
builder.add_node("logic_solver", logic_solver)
builder.add_node("result_interpreter", result_interpreter)
```

Here, we need to set conditional edges wherever output is based on the decision is based on the output from a node. We then call the compile function on builder to execute the program.

```Python
builder.set_entry_point("problem_classifier")
builder.add_conditional_edges(
    "problem_classifier",
    lambda state: state["classifier"],
    {
        "Logic Program": "logic_solver",
        "Constraint Satisfaction Problem": "constraint_solver",
        "Cannot classify": END
    }
)
builder.add_edge("logic_solver", "result_interpreter")
graph = builder.compile()
```

High Level Diagram



Scenarios:

1. When Facts are not provided in the question but present in the RAG context

Input

```
Python
question:'''Monica, Rachel and Phoebe are Female
Ross, Chandler and Joey are male
Ross, Rachel, Monica, Chandler and Joey are all friends

Is the below following statements is true, false or uncertain?
Amy is rachel\'s sister'''
```

Output

```shell
Refined result true
The statement is true.
```

2. With contextual RAG and open ended question

Input

```textproto
question'''Monica, Rachel and Phoebe are Female
            Pheobe is friends with both Rachel and Monica
            Ross is Monica\'s brother

            Provide the correct answer to the below question
            Whose sister is Amy ?
'''
```

Output

```textproto
Chain of Thought Reasoning
REASONING:
1. We need to determine whose sister Amy is, which means we will look for a
sibling relationship involving Amy.
2. From the facts, we have a sibling relationship: sibling(amy, rachel).
3. This indicates that Amy is a sister to Rachel.
4. Therefore, when we query sibling(amy, X), it matches with X being Rachel,
since Amy is Rachel's sister based on our fact.
5. Hence, the answer to the query "?- sibling(amy, X)." would resolve to X =
rachel.


Determined Answer:
Refined result [{'X': 'rachel'}]
Query: sibling(amy, X).
[{'X': 'rachel'}]
```

3. Complex query with multiple outputs where some facts are from kb and others from context.

Input

```python
question:
'''Ross and Joey are best friends
            Pheobe is friends with both Rachel and Monica
            Ross is Monica\'s brother
            Describe all bestfriends of Joey
'''
```

```textproto
Chain of Thought Reasoning
REASONING:
1. To determine bestfriends, we need mutual friendship between persons.
2. According to the facts, joey is friends with chandler and ross.
3. Check if chandler is also friends with joey; this is confirmed in the facts.
4. Therefore, joey and chandler are best friends.
5. Check if ross is also friends with joey; the friendship is mutual according
to the facts.
6. Therefore, joey and ross are best friends.
7. No other facts suggest a mutual friendship involving joey.
8. Hence, the query returns both chandler and ross as best friends of joey.

Query: bestfriends(joey, BestFriend).
[{'BestFriend': 'chandler'}, {'BestFriend': 'ross'}]
```