LLMs and Semantic Faithfulness in FM

Milestone 2

Ana Pires (PG61130) Miguel Carvalho (PG61153) Renato Garcia (PG61542)

Motivação

"While large language models (LLMs) [...] demonstrate proficiency in semantic extraction, they still encounter difficulties in addressing the complexity, ambiguity, and logical depth of real-world industrial requirements."

Referência: *Automated Translation of Software Requirements to LTL via Hierarchical Semantics Decomposition Using LLMs* (Req2LTL)

Milestone 1: Contexto & Desafios

Desafios

- *Silent failure*
- Custos:
    - trabalho manual
    - formação especializada

Conceitos Base

- LTL (vs. CTL)
- NL2Spec
- *Pipeline* NL → LTL

Prediction: AI will make formal verification go mainstream (Kleppmann, 2025)

02

NL2Spec *vs* Req2LTL

nl2spec: Interactively Translating Unstructured Natural Language to Temporal Logics with Large Language Models

Matthias Cosler[2], Christopher Hahn[1], Daniel Mendoza[1], Frederik Schmitt[2], and Caroline Trippel[1]

[1] Stanford University, Stanford, CA, USA
hahn@cs.stanford.edu, dmendo@stanford.edu, trippel@stanford.edu
[2] CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
matthias.cosler@cispa.de, frederik.schmitt@cispa.de

Abstract. A rigorous formalization of desired system requirements is indispensable when performing any verification task. This often limits the application of verification techniques, as writing formal specifications is an error-prone and time-consuming manual task. To facilitate this, we present nl2spec, a framework for applying Large Language Models (LLMs) to derive formal specifications (in temporal logics) from unstructured natural language. In particular, we introduce a new methodology to detect and resolve the inherent ambiguity of system requirements in natural language: we utilize LLMs to map subformulas of the formalization back to the corresponding natural language fragments of the input. Users iteratively add, delete, and edit these sub-translations to amend

Bridging Natural Language and Formal Specification–
Automated Translation of Software Requirements to LTL via Hierarchical Semantics Decomposition Using LLMs

Zhi Ma[1], Cheng Wen[2], Zhexin Su[1], Xiao Liang[1], Cong Tian[1*], Shengchao Qin[1] and Mengfei Yang[3]
[1]School of Computer Science and Technology, Xidian University, Xi'an, China
[2]Guangzhou Institute of Technology of Xidian University, Guangzhou, China
[3]China Academy of Space Technology, Beijing, China
{mazhi, wencheng, qinshengchao}@xidian.edu.cn, {23031212487, xiaoliang}@stu.xidian.edu.cn,
ctian@mail.xidian.edu.cn, yangmf@bice.org.cn

***Abstract*—Automating the translation of natural language (NL) software requirements into formal specifications remains a critical challenge in scaling formal verification practices to industrial settings, particularly in safety-critical domains. Existing approaches, both rule-based and learning-based, face significant limitations. While large language models (LLMs) like GPT-4o demonstrate proficiency in semantic extraction, they still encounter difficulties in addressing the complexity, ambiguity, and logical depth of real-world industrial requirements. In this paper, we propose Req2LTL, a modular framework that bridges NL and Linear Temporal Logic (LTL) through a hierarchical intermediate representation called *OnionL*. Req2LTL leverages LLMs for semantic decomposition and combines them with deter-**

require extensive labeled datasets and often struggle to generalize beyond their training examples. Recent advancements in large language models (LLMs), such as GPT-4o [17], have shown potential in related domains like code generation and logical inference [11], [18]–[22], yet directly applying these models to complex NL-to-LTL translation tasks remains problematic due to the implicit temporal semantics, deeply nested logical structures, and context-specific constraints inherent in industrial requirements.

Three primary challenges limit the effectiveness of applying LLMs directly to this translation task. First, natural language

03

Modelo *Baseline*

NL → LTL com *feedback* iterativo

Pipeline:
NL → *Generator* → *Validator* → *Judger* → *loop*

- Validação sintática (JSON + LTL)
- Avaliação semântica (*Judger*)

Insight:
- *explanation* ≈ proto-representação intermédia

04

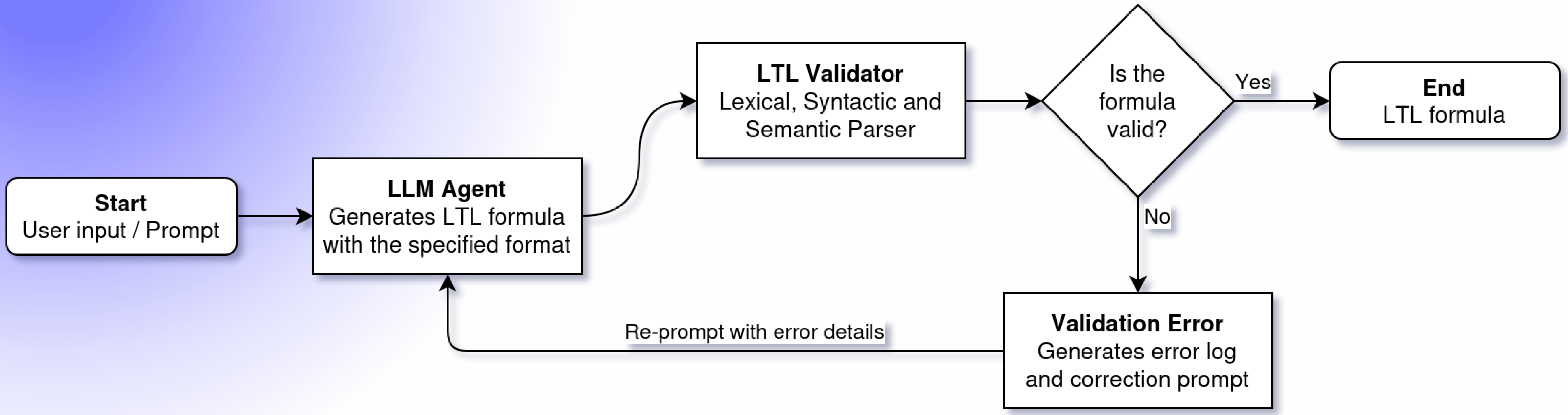

Diagrama 1: Etapa de validação de fórmulas LTL

05

06

Próximos Passos

- Representação Intermédia (IR)
- Avaliação e Métricas
- Extra…

Estrutura Intermédia

NL: "Every request is eventually
followed by a grant"

IR:

```
pattern  : response
scope    : global
trigger  : request
response : grant
```

LTL:

```
G (request -> F grant)
```

Referência: Dwyer patterns

Explanation (*insight* anterior):

```
request     : request
grant       : grant
globally    : G
eventually  : F
followed by : ->
```

07

Avaliação & Métricas

08

01.

NL ➔ Lifting

Extração semântica:

- proposições corretas
- operadores corretos

02.

Lifting ➔ LTL

Composição lógica

- estrutura da fórmula
- uso correto de operadores

03.

Grounded

Fidelidade semântica

- corresponde ao requisito?
- validado por humano/*judger*

09

Extra

Z3
SMT Solver

- Deteção de contradições entre requisitos
- Identificação de inconsistências lógicas
- Apoio à validação formal

Exemplo:

```
req1: G (a -> F b)
req2: G (a -> G ~b)
```

→ conflito condicionado

LLMs and Semantic Faithfulness in FM

Milestone 2

Ana Pires (PG61130)
Miguel Carvalho (PG61153)
Renato Garcia (PG61542)