LLMs and Semantic Faithfulness in FM

Milestone 3

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

01

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*)

Milestones anteriores: Contexto & Desafios

Desafios

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

Conceitos Base

- LTL (vs. CTL)
- NL2Spec
- Req2LTL
- *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-3 [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

CriticalSpec

University of Minho

Master's Degree in Software Engineering

Formal Methods of Programming Project

2025/2026

04

CriticalSpec: Semantic Faithfulness for LLM-Assisted Requirements Formalization

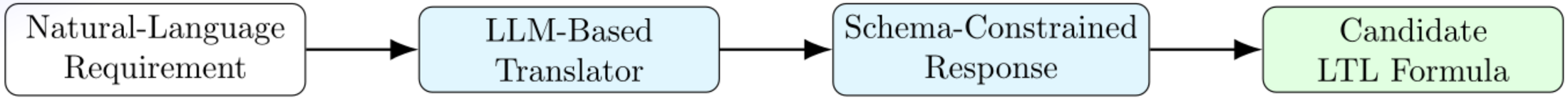
Natural-Language Requirements Translation to
Linear Temporal Logic for Safety-Critical Systems

05

Gestão de Agentes

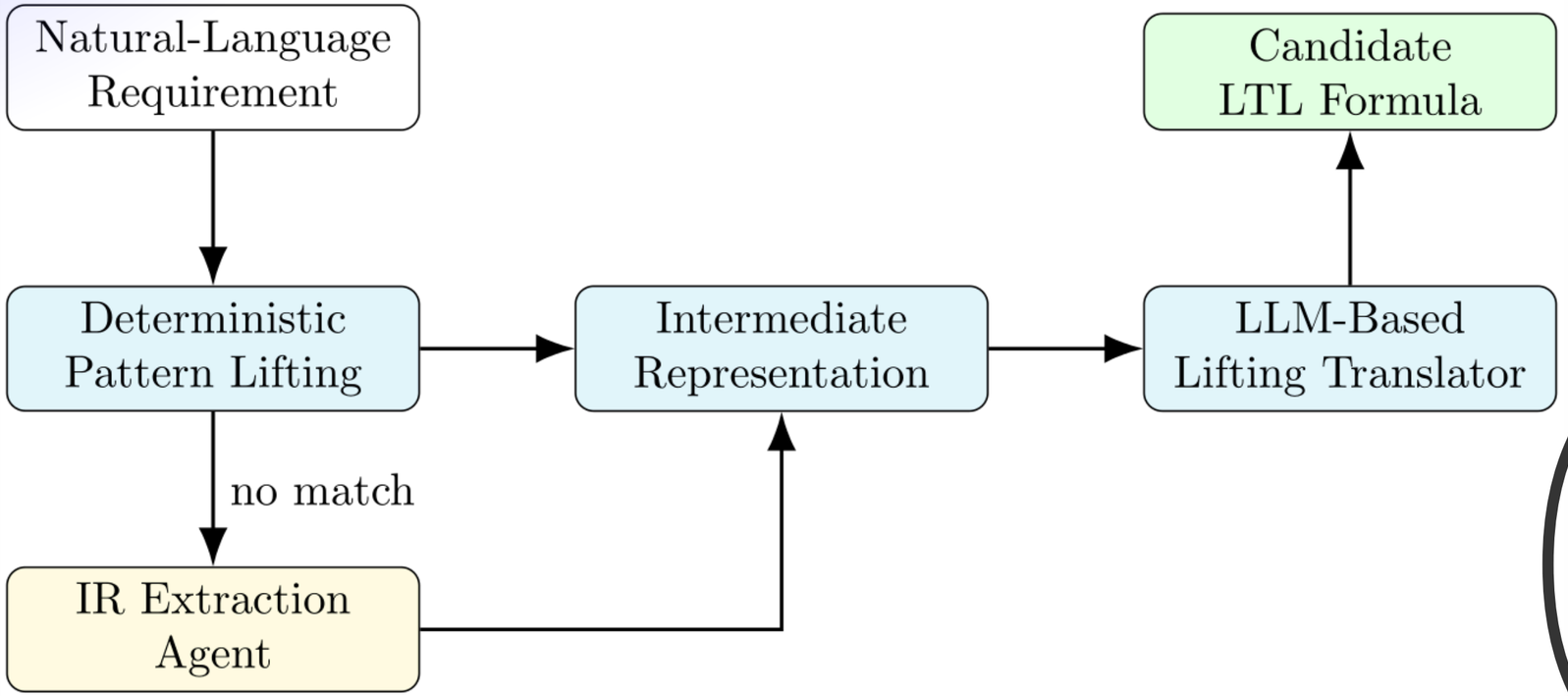
- Criar
- Remover
- Inspecionar detalhes
- Listar
- Limpar histórico de conversações

Direct NL-to-LTL Translation



06

Pattern-Based Lifting Translation



07

Estrutura Intermédia

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

IR:

```
pattern                : response
scope                  : global
trigger                : request
response               : grant
atomic propositions : [request, grant]
```

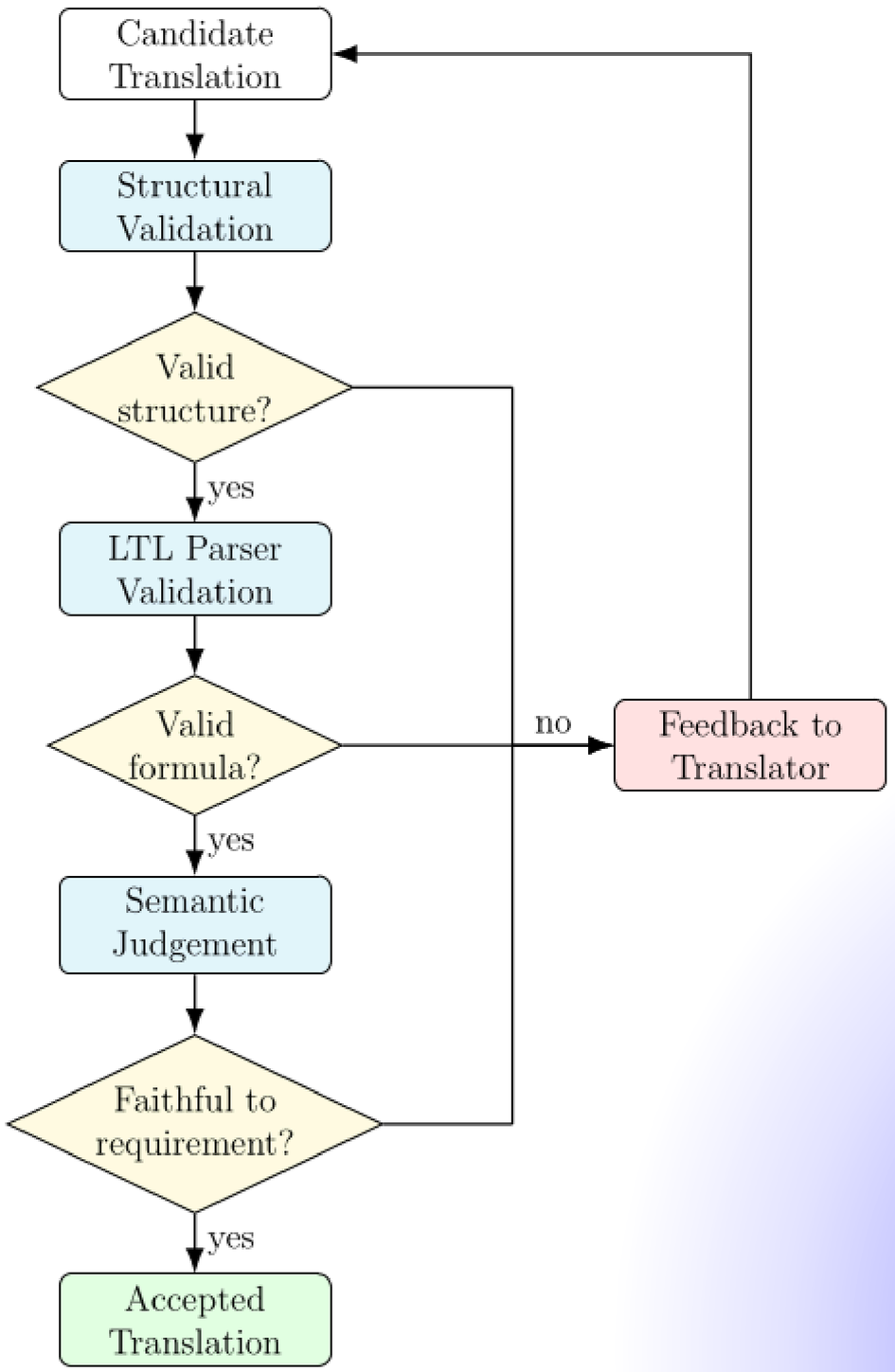
LTL:

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

Referência: *Dwyer patterns*

08

09 Validação



Avaliação & Métricas

01.

NL → Lifting

Extração semântica:

- proposições corretas
- operadores corretos

02.

Lifting → LTL

Composição lógica

- estrutura da fórmula
- uso correto de operadores

03.

Grounded

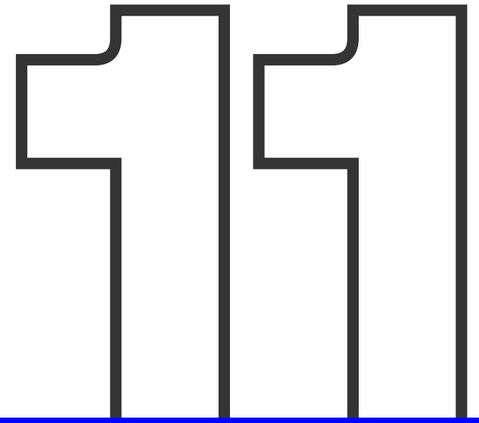
Fidelidade semântica

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

NL → Lifting

Avaliação da extração semântica de um requisito (NL)
para a representação intermédia (IR)

- Padrão (*pattern_match*)
- Âmbito (*scope_match*)
- Proposições atómicas (*f1_ap*)



Lifting → LTL

Avaliação da tradução da IR para LTL

- Estrutura lógica da fórmula
- Operadores temporais
- Preservação de relações
- Consistência entre condições e APs

Grounded

Avaliação final da fidelidade da fórmula gerada

Normalizações: Associatividade, e comutatividade, eliminação de negação dupla e implicações, leis de De Morgan e expansões de equivalência.

Bounded Semantic Relation (Lasso Checking)

$$\text{LTLScore} = \frac{\text{AnomMatch} + \text{AlphaMatch} + \text{CanonicalAnonMatch} + \text{CanonicalAlphaMatch}}{4}$$

12

13

extra

Z3
SMT Solver

- Deteção de contradições entre requisitos
- Apoio à validação formal pós-tradução

Exemplo:

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

→ conflito condicionado

Resultados

Experiência	Grounded	Lifted NL To TL	NL To Lifting
Lifting	86.96%	86.96%	84.85%
Lifting (No Judger)	26.09%	26.09%	61.53%
Lifting (gpt-5.4-mini)	47.83%	47.83%	63.21%
Direct	58.70%		
Direct (No Judger)	58.70%		

Conclusões

- *Pattern-based lifting* vs. *Direct translation*
- Importância do *judger*
- Avaliação e métricas

Trabalho Futuro

- *Deterministic pattern lifting* vs. *LLM-based IR extractor*
- Normalização NL
- Representação local de agentes

LLMs and Semantic Faithfulness in FM

Milestone 3

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