# Exploring the Role of WEST in Temporal Logic Validation:

# Strengths, Limitations, and Extensions

Songyan Lai

**Abstract**

This paper examines the strengths and limitations of using WEST for validating Mission-Time Linear Temporal Logic (MLTL) formulas. As a specialized tool, WEST leverages regular expressions to address a critical gap in the validation of MLTL formulas, particularly for finite and time-constrained systems. By comparing WEST with tools such as nuXmv, MLTLSAT, FRET, and R2U2, explore its functionality, efficiency, and areas for improvement. Recommendations for extending WEST's capabilities are also proposed to enhance its usability and integration into broader formal verification workflows.

## 1. Introduction

Temporal logic plays a critical role in specifying and verifying system requirements, particularly in mission-critical contexts where precise time constraints must be met. To facilitate these verifications, various tools have been developed, including nuXmv, MLTLSAT, FRET, R2U2, and WEST. Each of these tools is optimized for specific scenarios, providing unique validation techniques tailored to their respective supported logics.

This paper focuses on WEST, a tool developed to validate MLTL formulas using regular expressions. By generating finite timelines that satisfy temporal requirements, WEST offers an efficient solution for validating formulas in finite, time-constrained systems. Despite its advantages, WEST's reliance on strict syntax requirements and offline operation presents challenges for broader usability. The purpose of this paper is to evaluate WEST's functionality and propose enhancements that can address its current limitations while integrating it more effectively into the formal verification ecosystem.

## 2. Overview of Validation Tools

This section compares commonly used tools for temporal logic validation, highlighting their supported logics and approaches.

### 2.1 nuXmv

nuXmv is a symbolic model checker designed for verifying temporal logic requirements in diverse domains such as aerospace and automotive industries (Cimatti et al., 2014). While it excels in exhaustive state exploration, it is not optimized for MLTL formula validation.

- **Supported Temporal Logics**:
  - Linear Temporal Logic (LTL)
  - Computational Tree Logic (CTL)
  - Extended CTL (CTL*)
- **Validation Approaches**:
  - Model Checking: Verifies compliance by exploring all possible system states.
  - Counterexample Generation: Provides detailed feedback on failed validations.

### 2.2 MLTLSAT

MLTLSAT is a satisfiability solver tailored for MLTL. It encodes MLTL formulas into SAT problems for efficient satisfiability checking (Rozier, 2020). While effective for feasibility analysis, it does not directly support formula validation.

- **Supported Temporal Logic**:
  - Mission-Time Linear Temporal Logic (MLTL)
- **Validation Approach**:
  - SAT Solving: Determines whether an MLTL formula is satisfiable using advanced SAT-solving techniques.

### 2.3 FRET

FRET simplifies the formalization of natural language requirements into temporal logic (Giannakopoulou et al., 2020). It primarily focuses on requirement specification rather than validation.

- **Supported Temporal Logics**:
  - Linear Temporal Logic (LTL)
  - Metric Temporal Logic (MTL)
- **Validation Approaches**:
  - Requirement Formalization: Converts natural language inputs into temporal logic.
  - Consistency Checking: Ensures logical soundness of formalized requirements.

### 2.4 R2U2

R2U2 is a runtime verification tool designed for real-time monitoring of autonomous systems (Stark et al., 2018).

- **Supported Temporal Logics**:
  - Linear Temporal Logic (LTL)
  - Metric Temporal Logic (MTL)
  - Mission-Time Linear Temporal Logic (MLTL)
- **Validation Approaches**:
  - Real-Time Monitoring: Observes system behavior for compliance during operation.
  - Anomaly Detection: Identifies deviations in real-time.

### 2.5 WEST

WEST stands out by directly validating MLTL formulas through regular expressions. It generates finite timelines satisfying the formulas, akin to truth tables for propositional logic (Elwing et al., 2023).

- **Supported Temporal Logic**:
  - Mission-Time Linear Temporal Logic (MLTL)
- **Validation Approach**:
  - Regular Expression-Based Validation: Constructs finite timelines that meet formula specifications.

## 3. Key Features of WEST

### 3.1 Regular Expression-Based Validation

WEST's core innovation is its use of regular expressions for validating MLTL formulas. This approach ensures both soundness and completeness, producing detailed results for finite timelines. However, its reliance on regular expressions limits its applicability to formulas with complex dependencies.

### 3.2 Automation and Syntax Restrictions

WEST's syntax imposes stringent requirements, mandating that task-specific variables be represented as generic propositions like "p1" or "p2." This abstraction complicates input preparation, increasing the risk of user errors. Furthermore, WEST operates offline, which restricts its ability to provide dynamic feedback, unlike runtime tools like R2U2.

## 4. Extension of WEST

While WEST provides a solid foundation for MLTL validation, several enhancements could make it more user-friendly and versatile.

### 4.1 Accepting Descriptive Variable Names

WEST currently requires variables to be abstractly formatted, which adds complexity for users. A proposed improvement is enabling it to accept descriptive task-specific variables directly (e.g., "Task_A" or "Sensor_1"). By eliminating the need for manual abstraction, this change would make the tool more intuitive and reduce errors.

### 4.2 Refining the GUI

Although WEST features a graphical user interface, it is basic and lacks error-analysis capabilities. Upgrading the GUI could include:

- **Real-Time Error Detection**: Highlighting and explaining input errors with actionable feedback.
- **Enhanced Visualizations**: Providing clearer representations of formulas and validation timelines.

### 4.3 Integration with FRET

FRET's ability to formalize natural language requirements complements WEST's validation capabilities. An interface could be developed to translate FRET-generated LTL formulas into WEST-compatible MLTL syntax. Currently, I am working on a program that automatically converts task-specific variables (e.g., "Task_A") into WEST-compatible formats, bridging the gap between requirement specification and formula validation. This development underscores the potential for tighter integration between tools.

### 4.4 Expanding Logic Support

Expanding WEST's capabilities to support other temporal logics, such as LTL and MTL, would broaden its applicability to systems requiring continuous-time specifications.

## 5. Conclusion

WEST provides a focused solution for validating MLTL formulas, addressing a critical gap in formal methods. Its regular expression-based approach ensures accurate validation for finite timelines, but its strict syntax requirements and limited interface reduce usability. Enhancements such as supporting descriptive variable names, refining the GUI, and integrating with tools like FRET would make WEST more accessible and versatile. These improvements, along with expanded logic support, would strengthen WEST's position within the ecosystem of temporal logic validation tools, making it an invaluable resource for mission-critical applications.

## References


- Cimatti, A., Griggio, A., Tonetta, S., & Trentin, P. (2014). nuXmv: An SMT-based Model Checker. *International Conference on Computer-Aided Verification*.
- Elwing, J., Gamboa-Guzman, L., Sorkin, J., Travesset, C., Wang, Z., & Rozier, K.Y. (2023). Mission-Time LTL (MLTL) Formula Validation Via Regular Expressions. *International Conference on Temporal Logic*.
- Federal Aviation Administration (FAA) (2023). Formal Methods for Flight Certification. *FAA Technical Reports*.
- Giannakopoulou, D., Pasareanu, C., & Whalen, M. (2020). FRET: Formal Requirements Elicitation Tool. *NASA Formal Methods Symposium*.
- Rozier, K.Y. (2020). Mission-Time Linear Temporal Logic for Autonomous Systems. *IEEE Transactions on Software Engineering*.
- Stark, J., Rozier, K.Y., & Ivanov, I. (2018). R2U2: Monitoring and Health Management for Autonomous Systems. *International Conference on Runtime Verification*.