
**Faculty of Computer Systems & Software Engineering**

# Formal methods.

## Using TLC Model Checker

**Vitaliy Mezhuyev**

# Introduction to TLC

TLC handles specifications in the standard form

$$Init \wedge \Box[Next]_{vars} \wedge Temporal$$

where
***Init*** is the initial predicate
***Next*** is the next-state action
***vars*** is the tuple of all model variables
***Temporal*** is a temporal formula that specifies a *liveness* condition.

# Introduction to TLC

TLC works by generating behaviors that satisfy the specification.

To do this, should be formed a **model** of the specification.

To define a model, first we must assign values to the specification's constant parameters. E.g. for AsyncInteface

**CONSTANT** Data = {d1, d2, d3}

# Modes of TLC

There are two ways to use TLC. The default method is model checking, in which it tries to find all reachable states (that is, all states that can occur in behaviors satisfying the formula).

You can also run TLC in simulation mode, in which it randomly generates behaviors, without trying to check all reachable states.

Second way is useful then we have huge amount of states.

# TLC input

The input to TLC consists of a TLA module and configuration. The configuration tells TLC the names of the specification and of the properties to be checked.

For example for HourClock we need specify behavior as:

SPECIFICATION **HC**
OR Initial and next state predicates**:**
**HCini and Hcnext**

SPECIFICATION **HC** statement tells TLC that HC is the specification that it should check.

# Properties to be checked

For HourClock.tla TLC check the **invariant** property **HCini**
Here invariant **HCini** specifies a state predicate.

In other words, TLC checks that formula **HCini** is an invariant of the specification **HC**, or, that the specification implies []HCini.

**THEOREM  HC => []HCini**

TypeInvariance == []HCini
This temporal formula asserting that HCini is always true
**THEOREM  HC => TypeInvariance**

For **LiveHourClock** we will introduce liveness properties
PROPERTIES AlwaysTick AllTimes TypeInvariance
THEOREM  LSpec => AlwaysTick /\ AllTimes /\ TypeInvariance

# Types of errors to be checked

To find errors in a specification is to verify that it satisfies properties that it should.

You can also run TLC without having it check any property, in which case it will just look for two kinds of errors:

- “Silliness" errors. A silly expression is one like “3 + <<1”, “2>”, whose meaning is not determined by the semantics of TLA

- Deadlock. it is expressed by the invariance property $\Box(\text{ENABLED } Next)$

# TLC Values

TLC can compute only a restricted class of values, called TLC values. Those values are built from the following four types of primitive values:

| | |
|---|---|
| Booleans | Values true and false. |
| Integers | Values like 123. |
| Strings | Values like “abc". |
| Model Values | Values introduced in the CONSTANT statement, e.g. {d1, d2, d3} |

# How TLC Evaluates Expressions

TLC evaluates expressions in a straightforward way, generally evaluating subexpressions “from left to right". In particular:

- It evaluates $p \land q$ by first evaluating $p$ and, if it equals TRUE, then evaluating $q$.
- It evaluates $p \lor q$ by first evaluating $p$ and, if it equals FALSE, then evaluating $q$. It evaluates $p \Rightarrow q$ as $\neg p \lor q$.
- It evaluates IF $p$ THEN $e_1$ ELSE $e_2$ by first evaluating $p$, then evaluating either $e_1$ or $e_2$.

**Thank you for your attention!**
**Please ask questions**