

---

# **SRC Technical Note**

## **1994 - 001**

**December 16, 1994**

---

# **Introduction to TLA**

Leslie Lamport



**Systems Research Center**

130 Lytton Avenue

Palo Alto, California 94301

<http://www.research.digital.com/SRC/>

---

## A Simple Example

We begin by specifying a system that starts with  $x$  equal to 0 and keeps incrementing  $x$  by 1 forever. In a conventional programming language, this might be written

```
initially  $x = 0$  ;  
loop forever  $x := x + 1$  end loop
```

The TLA specification is a formula  $\Pi$  defined as follows, where the meaning of each conjunct is indicated by the comments.

$$\begin{aligned}\Pi &\triangleq (x = 0) && \text{Initially, } x \text{ equals 0.} \\ &\quad \wedge \Box[x' = x + 1]_x && \text{Always } (\Box), \text{ the value of } x \text{ in the next} \\ &&& \text{state } (x') \text{ equals its value in the current} \\ &&& \text{state } (x) \text{ plus 1. Ignore the subscript } x \\ &&& \text{for now.} \\ &\quad \wedge \text{WF}_x(x' = x + 1) && \text{Ignore this for now.}\end{aligned}$$

As specifications get more complicated, we need better methods of writing formulas. We use lists of formulas bulleted with  $\wedge$  and  $\vee$  to denote conjunctions and disjunctions, and we use indentation to eliminate parentheses. The definition of  $\Pi$  can then be written as

$$\begin{aligned}\Pi &\triangleq \wedge x = 0 \\ &\quad \wedge \Box[x' = x + 1]_x \\ &\quad \wedge \text{WF}_x(x' = x + 1)\end{aligned}$$

## What a Formula Means

A TLA formula is true or false on a *behavior*, which is a sequence of *states*, where a state is an assignment of values to variables. Formula  $\Pi$  is true on a behavior in which the  $i^{\text{th}}$  state assigns the value  $i - 1$  to  $x$ , for  $i = 1, 2, \dots$

Systems are real; behaviors are mathematical objects. To decide if a system  $S$  satisfies formula  $\Pi$ , we must first have a way of representing an execution of  $S$  as a behavior (a sequence of states). Given such a representation, we say that system  $S$  satisfies formula  $\Pi$  (or that  $S$  implements the specification  $\Pi$ ) iff (if and only if)  $\Pi$  is true for every behavior corresponding to a possible execution of  $S$ .

## Another Example

Next, we specify a system that starts with  $x$  and  $y$  both equal to 0 and repeatedly increments  $x$  and  $y$  by 1. A step increments either  $x$  or  $y$  (but not both). The variables are incremented in arbitrary order, but each is incremented infinitely often. This system might be represented in a conventional programming language as

```

initially   $x = 0, y = 0$  ;
cobegin
  loop forever   $x := x + 1$   end loop ||
  loop forever   $y := y + 1$   end loop
coend

```

The TLA specification is the formula  $\Phi$ , defined as follows. For convenience, we first define two formulas  $\mathcal{X}$  and  $\mathcal{Y}$ , and then define  $\Phi$  in terms of  $\mathcal{X}$  and  $\mathcal{Y}$ .

$\mathcal{X} \triangleq \wedge x' = x + 1$     An  $\mathcal{X}$  step is one that increments  $x$   
 $\wedge y' = y$             and leaves  $y$  unchanged.

$\mathcal{Y} \triangleq \wedge y' = y + 1$     A  $\mathcal{Y}$  step is one that increments  $y$   
 $\wedge x' = x$             and leaves  $x$  unchanged.

$\Phi \triangleq \wedge (x = 0) \wedge (y = 0)$             Initially,  $x$  and  $y$  equal 0.  
 $\wedge \Box[\mathcal{X} \vee \mathcal{Y}]_{\langle x, y \rangle}$             Every step is either an  $\mathcal{X}$  step or a  $\mathcal{Y}$  step.  
 $\wedge \text{WF}_{\langle x, y \rangle}(\mathcal{X}) \wedge \text{WF}_{\langle x, y \rangle}(\mathcal{Y})$     As explained later, this asserts that infinitely many  $\mathcal{X}$  and  $\mathcal{Y}$  steps occur.

Formulas  $\mathcal{X}$  and  $\mathcal{Y}$  are called *actions*. An action is true or false on a *step*, which is a pair of states—an old state, described by unprimed variables, and a new state, described by primed variables.

## Implementation and Stuttering

We say that a specification (TLA formula)  $F$  implements a specification  $G$  iff every system that satisfies  $F$  also satisfies  $G$ . This is true if every behavior that satisfies  $F$  also satisfies  $G$ , which means that all behaviors satisfy the formula  $F \Rightarrow G$ . A formula is said to be *valid* iff it is satisfied by all behaviors. (“All behaviors” means all sequences of states, not just ones that represent the execution of some particular system.) So,  $F$  implements  $G$  if the formula  $F \Rightarrow G$  is valid. Implementation is implication.

A system that repeatedly increments  $x$  and  $y$  repeatedly increments  $x$ . Therefore, specification  $\Phi$  should implement specification  $\Pi$ . This means that every behavior satisfying  $\Phi$  should also satisfy  $\Pi$ . Behaviors that satisfy  $\Phi$  allow steps that increment  $y$  and leave  $x$  unchanged. Therefore,  $\Pi$  must allow steps that leave  $x$  unchanged. That's where the subscript  $x$  comes in. For any action (Boolean formula containing constants, variables and primed variables)  $\mathcal{A}$  and every state function (expression containing only constants and unprimed variables)  $f$ , we define

$$[\mathcal{A}]_f \triangleq \mathcal{A} \vee (f' = f)$$

where  $f'$  is the expression obtained by priming all the variables in  $f$ . Thus, a step satisfies  $[\mathcal{A}]_f$  iff it satisfies  $\mathcal{A}$  or it leaves  $f$  unchanged. The formula  $\Box[\mathcal{A}]_f$  asserts that every step is an  $\mathcal{A}$  step (one that satisfies  $\mathcal{A}$ ) or leaves  $f$  unchanged. Hence, the conjunct  $\Box[x' = x + 1]_x$  of  $\Pi$  does allow steps that leave  $x$  unchanged. Such steps are called *stuttering* steps.

In mathematics, the formula  $x^2 = x + 1$  is not an assertion about a universe just containing  $x$ ; it is an assertion about a universe containing all possible variables, including  $x$ ,  $y$ , and  $z$ . The formula  $x^2 = x + 1$  simply doesn't say anything about  $y$  and  $z$ . Similarly, formula  $\Pi$  is an assertion about sequences of states, where a state is an assignment of values to all variables, not just to  $x$ . Formula  $\Pi$  specifies a system whose execution is described by the changes to  $x$ . But a behavior represents a history of some entire universe containing that system. To be a sensible specification,  $\Pi$  must allow stuttering steps in which other parts of the universe change while  $x$  remains unchanged.

Similarly,  $\Phi$  allows steps that leave the pair  $\langle x, y \rangle$  unchanged, and therefore leave both  $x$  and  $y$  unchanged. If we are just observing  $x$  and  $y$ , then there is no way to tell that such a step has occurred.

Stuttering steps make it unnecessary to consider finite behaviors. An execution in which a system halts is represented by an infinite behavior in which the variables describing that system stop changing after a finite number of steps. When a system halts, it doesn't mean that the entire universe comes to an end. Thus, by a behavior, we mean an infinite sequence of states.

## Fairness

Formula  $\Box[x' = x + 1]_x$  allows arbitrarily many steps that leave  $x$  unchanged. In fact, it is satisfied by a behavior in which  $x$  never changes. We want to

require that  $x$  be incremented infinitely many times, so our specification must rule out behaviors in which  $x$  is incremented only a finite number of times. This is accomplished by the WF formula, as we now explain.

An action  $\mathcal{A}$  is said to be *enabled* in a state  $s$  iff there exists some state  $t$  such that the pair of states  $\langle \text{old-state } s, \text{new-state } t \rangle$  satisfies  $\mathcal{A}$ . The formula  $\text{WF}_f(\mathcal{A})$  asserts of a behavior that, if the action  $\mathcal{A} \wedge (f' \neq f)$  ever becomes enabled and remains enabled forever, then infinitely many  $\mathcal{A} \wedge (f' \neq f)$  steps occur. In other words, if it ever becomes possible and remains forever possible to execute an  $\mathcal{A}$  step that changes  $f$ , then infinitely many such steps must occur.

Any integer can be incremented by 1 to produce a different integer. Hence, the action  $(x' = x + 1) \wedge (x' \neq x)$  is enabled in any state where  $x$  is an integer. The formula  $(x = 0) \wedge \Box[x' = x + 1]_x$ , which asserts that  $x$  is initially 0 and in every step is either incremented by 1 or left unchanged, implies that  $x$  is always an integer. Hence, this formula implies that  $(x' = x + 1) \wedge (x' \neq x)$  is always enabled. Hence, the conjunct  $\text{WF}_x(x' = x + 1)$  of  $\Pi$  asserts that infinitely many  $(x' = x + 1) \wedge (x' \neq x)$  steps occur. Hence,  $\Pi$  asserts that  $x$  is incremented infinitely often, as desired.

Similarly,  $(x = 0) \wedge \Box[\mathcal{X} \vee \mathcal{Y}]_{\langle x, y \rangle}$  implies that  $x$  is always an integer, so  $\mathcal{X} \wedge (\langle x, y' \rangle \neq \langle x, y \rangle)$  is always enabled. Hence,  $\Phi$  implies that  $x$  is incremented infinitely often. Every behavior satisfying  $\Phi$  does satisfy  $\Pi$ , so  $\Phi \Rightarrow \Pi$  is valid.

WF stands for *Weak Fairness*. TLA specifications also use *Strong Fairness* formulas of the form  $\text{SF}_f(\mathcal{A})$ , where  $f$  is a state function and  $\mathcal{A}$  an action. This formula asserts that if  $\mathcal{A} \wedge (f' \neq f)$  is enabled infinitely often (in infinitely many states of the behavior), then infinitely many  $\mathcal{A} \wedge (f' \neq f)$  steps must occur. If an action ever becomes enabled forever, then it is enabled infinitely often. Hence,  $\text{SF}_f(\mathcal{A})$  implies  $\text{WF}_f(\mathcal{A})$ ; strong fairness implies weak fairness.

The subscripts in WF and SF formulas (and in the formula  $\Box[\mathcal{N}]_f$ ) make it syntactically impossible to write a formula that can distinguish whether or not stuttering steps have occurred. In practice, whenever we write a formula of the form  $\text{WF}_f(\mathcal{A})$  or  $\text{SF}_f(\mathcal{A})$ , action  $\mathcal{A}$  will imply  $f' \neq f$ , so any  $\mathcal{A}$  step changes  $f$ .

## Hiding

The formula  $\exists y : \Phi$  is satisfied by a behavior iff there is some sequence of values that can be assigned to  $y$  which would produce a behavior satisfying  $\Phi$ . (This definition is only approximately correct; see [2] for the precise definition.) The temporal existential quantifier  $\exists y$  is the formal expression of what it means to “hide” the variable  $y$  in a specification. If we hide  $y$  in a specification asserting that  $x$  and  $y$  are repeatedly incremented, we get a specification asserting that  $x$  is repeatedly incremented. Thus, the specification obtained by hiding  $y$  in  $\Phi$  should be equivalent to  $\Pi$ . Indeed, the formula  $\exists y : \Phi$  is equivalent to  $\Pi$ . In other words, the formula  $(\exists y : \Phi) \equiv \Pi$  is valid.

## Composition

Let  $\mathcal{X}$  and  $\mathcal{Y}$  be the actions defined above, and let

$$\begin{aligned}\Pi_x &\triangleq (x = 0) \wedge \Box[\mathcal{X}]_x \wedge \text{WF}_{\langle x, y \rangle}(\mathcal{X}) \\ \Pi_y &\triangleq (y = 0) \wedge \Box[\mathcal{Y}]_y \wedge \text{WF}_{\langle x, y \rangle}(\mathcal{Y})\end{aligned}$$

A simple calculation shows that, if  $x$  and  $y$  are integers, then  $[\mathcal{X}]_x \wedge [\mathcal{Y}]_y$  is equivalent to  $[\mathcal{X} \vee \mathcal{Y}]_{\langle x, y \rangle}$ . It follows from this and the laws of temporal logic that  $\Pi_x \wedge \Pi_y$  is equivalent to  $\Phi$ . We can interpret  $\Pi_x$  and  $\Pi_y$  as the specifications of two processes, one repeatedly incrementing  $x$  and the other repeatedly incrementing  $y$ , in a program whose variables are  $x$  and  $y$ . Composing two such processes yields a program, with variables  $x$  and  $y$ , that repeatedly increments both  $x$  and  $y$ —the program specified by  $\Phi$ .

In general, a specification  $F$  of a system  $S$  describes the behaviors (representing histories) of a universe in which  $S$  operates correctly. A specification  $G$  of a system  $T$  describes behaviors of the same universe in which  $T$  operates correctly. Composing  $S$  and  $T$  means ensuring that both  $S$  and  $T$  operate correctly in that universe. The behaviors of a universe in which both systems operate correctly are described by the formula  $F \wedge G$ . Composition is conjunction.

## Assumption/Guarantee Specifications

An assumption/guarantee specification asserts that a system operates correctly if the environment does. Let  $M$  be a formula asserting that the sys-

tem does what we want it to, and let  $E$  be a formula asserting that the environment does what it is supposed to. We would expect the assumption/guarantee specification to be  $E \Rightarrow M$ , the formula asserting that either  $M$  is satisfied (the system behaved as desired) or  $E$  is not satisfied (the environment did not behave correctly). However, we instead write the stronger specification  $E \pmtriangleright M$ , which asserts both that  $E$  implies  $M$ , and that no step can make  $M$  false unless  $E$  has already been made false. The precise meaning of the formula  $E \pmtriangleright M$  is given in [1].

## All of TLA

TLA is built on a logic of actions, which is a language for writing predicates, state functions, and actions, and a logic for reasoning about them. A predicate is a Boolean expression containing constants and variables; a state function is a nonBoolean expression containing constants and variables; and an action is a Boolean expression containing constants, variables, and primed variables. The complete specification language  $\text{TLA}^+$ , described elsewhere, includes such a language.

Syntactically, a TLA formula has one of the following forms:

$$\begin{array}{lllll}
 P & \Box[\mathcal{A}]_f & \Box F & \exists x : F & \\
 \neg F & F \wedge G & F \vee G & F \Rightarrow G & F \equiv G \\
 \text{WF}_f(\mathcal{A}) & \text{SF}_f(\mathcal{A}) & F \pmtriangleright G & \Diamond F & F \leadsto G
 \end{array}$$

where  $P$  is a predicate,  $f$  is a state function,  $\mathcal{A}$  is an action,  $x$  is a variable, and  $F$  and  $G$  are TLA formulas. The last row of formulas can be expressed in terms of the others (and of course, all the Boolean operators can be defined from  $\neg$  and  $\wedge$ ). The Boolean operators have their usual meanings; the meanings of the other operators are described below.

- $P$  Satisfied by a behavior iff  $P$  is true for (the values assigned to variables by) the initial state.
- $\Box[\mathcal{A}]_f$  Satisfied by a behavior iff every step satisfies  $\mathcal{A}$  or leaves  $f$  unchanged.
- $\Box F$  ( $F$  is always true.) Satisfied by a behavior iff  $F$  is true for all suffixes of the behavior.

- $\exists x : F$  Satisfied by a behavior iff there are some values that can be assigned to  $x$  to produce a behavior satisfying  $F$ . (See [2] for the precise definition.)
- $\text{WF}_f(\mathcal{A})$  (Weak fairness of  $\mathcal{A}$ ) Satisfied by a behavior iff  $\mathcal{A} \wedge (f' \neq f)$  is infinitely often not enabled, or infinitely many  $\mathcal{A} \wedge (f' \neq f)$  steps occur.
- $\text{SF}_f(\mathcal{A})$  (Strong fairness of  $\mathcal{A}$ ) Satisfied by a behavior iff  $\mathcal{A} \wedge (f' \neq f)$  is only finitely often enabled, or infinitely many  $\mathcal{A} \wedge (f' \neq f)$  steps occur.
- $F \triangleleft G$  Is true for a behavior iff  $G$  is true for at least as long as  $F$  is. (See [1] for the precise definition.)
- $\Diamond F$  ( $F$  is eventually true) Defined to be  $\neg \Box \neg F$ .
- $F \leadsto G$  (Whenever  $F$  is true,  $G$  will eventually become true) Defined to be  $\Box(F \Rightarrow \Diamond G)$ .

## References

- [1] Martín Abadi and Leslie Lamport. Conjoining specifications. *ACM Transactions on Programming Languages and Systems*, 17(3):507–534, May 1995.
- [2] Leslie Lamport. The temporal logic of actions. *ACM Transactions on Programming Languages and Systems*, 16(3):872–923, May 1994.