Formalizing synthesis in TLA$^+$

Ioannis Filippidis and Richard M. Murray

Control and Dynamical Systems
California Institute of Technology
{ifilippi,murray}@caltech.edu

December 23, 2016

Abstract

This report proposes a TLA$^+$ definition for the problem of constructing a strategy that implements a temporal property. It is based on a note by Lamport [1] that outlines a formalization of realizability in TLA. The modified definition proposed here is expressed axiomatically in TLA$^+$. Specifying what function is acceptable as a strategy requires care, so that a function with empty domain be avoided, while ensuring that the strategy will not need to have a domain too large to be a set.

We prove that initial conditions should appear in assumptions only, unless an initial predicate is added to the definition of a realization. We show that a specification should include an assumption about a set of initial values to ensure that realizability does not become unprovable. We discuss what form of open-system properties expressed with the "while-plus" operator $\overset{+}{\rhd}$ are realizable.

We formalize the notions of interleaving and disjoint-state behaviors, based on definitions given by Lamport and Abadi, and consider the notion of interleaving for an open-system property. We give examples of expressing different forms of games in TLA$^+$ using the proposed definition, including games with partial information.

Contents

1 Motivation

A TLA$^+$ formula φ describes some behaviors[1]. Constructing an implementation is a separate concern. In TLA$^+$, implementation is implication between two properties $\psi \Rightarrow \varphi$. Synthesis is the construction of an implementation ψ that we can run on a computer. A computer only knows how to execute some concrete steps, so the synthesized property ψ should be expressed in terms of these steps.

In order to define synthesis, we must describe what our computer can do, for example what variables it can change. A specification for the synthesis problem describes precisely what it means to control a variable. Formalizing the notion of variable controllability was one motivation for what follows.

Another reason was the observation that, in the literature on games, a strategy is sometimes allowed to be a *partial* function. The intention is that a smaller domain may suffice to define a winning strategy. However, such a definition can leave unspecified what happens outside the strategy's domain[2]. If plays outside the strategy's domain are declared as impossible due to typeness considerations, then a function with empty domain would qualify as a winning strategy[3]. The definition of synthesis formalized below in TLA$^+$ aims to avoid such ambiguity, a symptom of using a typed logic [5, 6, 7].

2 Functions as strategies

We want to define the solution of a game in TLA$^+$. Several other definitions of this problem exist in the literature. Based on the definition outlined in [1], we give a TLA$^+$ definition that is general enough to serve for more than one kind of game, including games of partial information.

We have to choose what qualifies as a strategy. Is an operator a strategy? A strategy reads the current state and writes (part of) the next state. So, a strategy operator must take arguments. To express realizability, we will want to quantify over system strategies. TLA$^+$ is a first-order logic, so we cannot quantify over unary operators [8, p.318], [5, p.508]. In order to quantify over strategies, a strategy should be a set. We want to use a strategy as a mapping. There are several alternative ways for defining sets that can serve the purpose of mappings [6, p.4]. We prefer to let a strategy be a function f, similarly to [1, 9, 10]. This requirement can be expressed axiomatically in TLA$^+$ with the predicate

$$IsAFunction(f) \;\triangleq\; f = [x \in \textsc{domain}\ f \mapsto f[x]]$$

[8, p.303]. This is convenient, because it does not mention a set of candidate strategies, and leads to a simpler definition of synthesis below. This predicate

[1] Any formula describes a collection of behaviors too large to be a set (a proper class).

[2] A partial function f can lead the game outside its domain, artificially terminating it.

[3] Non-axiomatic definitions of synthesis that avoid this have been given using partial functions [2, p.46], [3, p.367], [4, p.915]. They are informal, and more complex, because trivial solutions need to be expressly excluded.

can be written equivalently as

$$\mathit{IsAFunction}(f) = \exists\, S \; : \; \begin{array}{l} \land\; S = \textsc{domain}\, f \\ \land\; f = [x \in S \mapsto f[x]]. \end{array}$$

We are about to *construct* a function f suitable for our purpose, and so choose DOMAIN f. This choice raises the question of what specifications admit a function as strategy.

3 Specifications that functions can implement

3.1 Initial conditions

Somewhat simplified, function f reads the variables x, h and controls the next value of x,

Definition 1 (Simple realization) *Let*

$$\psi(x, h, f) \;\triangleq\; \Box[\, x' = f[\langle x, h\rangle]\,]_{\langle x,h\rangle}.$$

The objective of this control law is to implement the property $\varphi(x, h)$ that mentions only the variables x and h, and does not mention the constant f. Proving that f implements the property $\varphi(x, h)$ (roughly) means proving that

$$\models \psi(x, h, f) \;\Rightarrow\; \varphi(x, h).$$

Above, we decided that a strategy should be a function f, so we want

$$\models \begin{array}{l} \land\; \mathit{IsAFunction}(f) \\ \land\; \psi(x, h, f) \;\Rightarrow\; \varphi(x, h). \end{array}$$

Any property φ implies an initial (state) predicate $\mathit{Init}(x, h)$, even if it is trivial (TRUE). We will show that for ψ to implement φ:

1. ψ should contain a (non-trivial) initial predicate, or
2. no initial state should violate φ.

For any initial value u of x, we will construct a behavior that satisfies ψ but violates φ. The proofs are in Appendix A. The proof style is described in [11]. The operator $\boldsymbol{\exists}$ expresses temporal existential quantification [8, §8.8, p.109].

Proposition 2
ASSUME: *1.* CONSTANTS u, f *(rigid variables)*
2. VARIABLE h *(flexible variable)*
PROVE: $\boldsymbol{\exists}\, x \; : \; \begin{array}{l} \land\; x = u \\ \land\; \Box[x' = f[\langle x, h\rangle]]_{\langle x,h\rangle} \end{array}$

In words, for any behavior[4] there exists some stutter-equivalent x-variant behavior that satisfies $(x = u) \wedge \Box[x' = f[\langle x, h\rangle]]_{\langle x,h\rangle}$. The proof is written at the meta-level, in Zermelo-Fraenkel (ZF) set theory [12]. For this reason, we translate the statement to ZF, using definitions from [8, p.316], and

$$(1) \qquad \forall_{\text{behavior}}\, \sigma : F \;\triangleq\; \neg\, \exists_{\text{behavior}}\, \sigma : \neg F = \forall\, \sigma : \mathit{IsABehavior}(\sigma) \Rightarrow F$$

Proposition 3

ASSUME: *1.* CONSTANTS u, f

2. VARIABLES h, x

PROVE: $\forall_{\text{behavior}}\, \sigma : \exists_{\text{behavior}}\, \tau : \begin{array}{l} \wedge\ \tau \sim_x \sigma \\ \wedge\ \tau \models \begin{array}{l} \wedge\ x = u \\ \wedge\ \Box[x' = f[\langle x, h\rangle]]_{\langle x,h\rangle} \end{array} \end{array}$

PROOF: on page 34.

[Proposition 3] PROOF SKETCH: Given any behavior σ and TLA$^+$ constant u, pick a behavior τ with[5]

$$\begin{array}{l} \tau[n \in \mathit{Nat}] \triangleq \text{IF } n = 0 \text{ THEN } [\sigma[0] \text{ EXCEPT } ![\![x]\!] = \sigma[0][\![u]\!]] \\ \qquad\qquad\qquad\qquad\quad \text{ELSE } [\sigma[n] \text{ EXCEPT } ![\![x]\!] = \tau[n-1][\![f[\langle x, h\rangle]]\!]] \end{array}$$

By construction, all steps[6] of τ satisfy the action $x' = f[\langle x, h\rangle]$, and initially $x = u$. This construction is possible because f and $\langle x, h\rangle$ are sets, and TLA$^+$ an untyped logic, so the expression $f[\langle x, h\rangle]$ is syntactically well-formed [8, p.71]. In an untyped logic, every syntactically well-formed expression is interpreted as some set [8, p.67]. So, the expression $f[\langle x, h\rangle]$ has some set as meaning.

Next, we show that if some initial states violate φ then no function f can implement φ with ψ.

Lemma 4

ASSUME: *1.* STATE $\mathit{Init}(x, h)$

2. TEMPORAL $\varphi(x, h)$

3. No variable symbols other than x, h occur in $\varphi(x, h)$

4. ψ as defined in Definition 1

5. $\varphi(x, h) \Rightarrow \mathit{Init}(x, h)$

6. $\exists\, u, r : \mathit{Init}(u, r) \equiv \text{FALSE}$

PROVE: $\forall f : \boldsymbol{\exists}\, x, h : \psi(x, h, f) \wedge \neg\varphi(x, h)$.

[4] In Proposition 2, the proof goal is equivalent to its universal closure [13, p.5].

[5] Using TLA$^+$ notation also for ZF, ! refers to the function being defined. The variable interpretation $\mathit{state}[\![u]\!]$ is defined in [8, p.311].

[6] A *step* of a behavior is defined in [8, p.16].

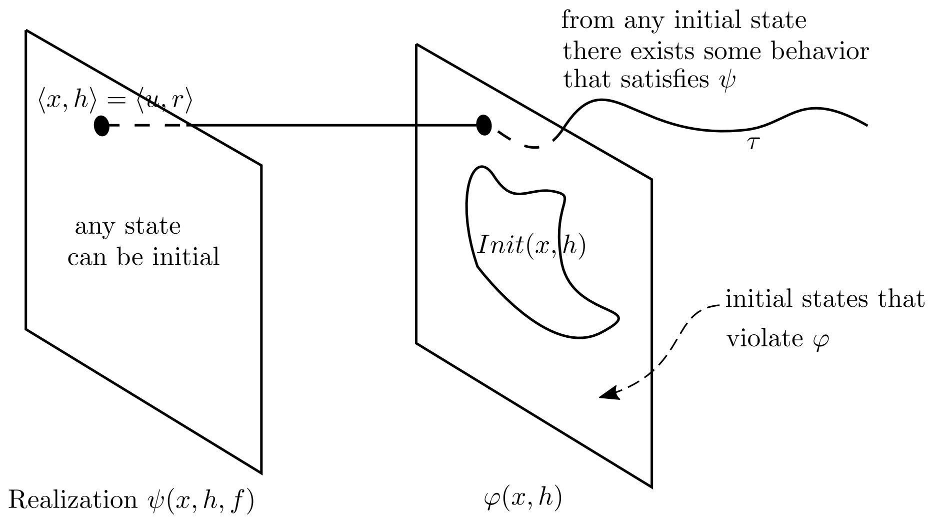

Figure 1: If some initial state violates φ, and no initial predicate I is included in the realization ψ, then φ is unrealizable.

Proof: on page 36.

[Lemma 4] Proof sketch: We can pick initial values u, r such that

$$(\langle x, h\rangle = \langle u, r\rangle) \;\Rightarrow\; (Init(x, h) \equiv \text{FALSE})$$

By contrapositive of assumption 5

$$\neg Init(x, h) \Rightarrow \neg\varphi(x, h)$$

So

$$(\langle x, h\rangle = \langle u, r\rangle) \;\Rightarrow\; \neg\varphi(x, h) \tag{2}$$

By Proposition 2, which holds $\forall\, h$ (universal closure)

$$\exists\, x, h \;:\; \begin{array}{l} \wedge\ \langle x, h\rangle = \langle u, r\rangle \\ \wedge\ \psi(x, h, f) \end{array} \tag{3}$$

By Eqs. (2) and (3), for any f

$$\begin{array}{l} \exists\, x, h \;:\; \begin{array}{l} \wedge\ \psi(x, h, f) \\ \wedge\ \langle x, h\rangle = \langle u, r\rangle \end{array} \quad \Rightarrow \\ \exists\, x, h \;:\; \psi(x, h, f) \wedge \neg\varphi(x, h). \end{array}$$

3.2 From where should a strategy matter?

Suppose that behaviors that assign arbitrary values to x, h can violate the property $\varphi(x, h)$. No function f in the property $\psi(x, h, f)$ can provably implement

$\varphi(x, h)$ in this case, because DOMAIN f must be a set. This suggests that φ should include an assumption of the form $\langle x, h\rangle \in S$ about the initial condition. We will prove that otherwise realizability can be impossible to prove. The proof relies on the freedom that exists in how function application, $f[x]$, is interpreted when $x \notin$ DOMAIN f. TLA$^+$ is untyped, so $f[x]$ is some set. The axioms of TLA$^+$ leave unspecified what set $f[x]$ is for $x \notin$ DOMAIN f.

3.2.1 Interpretation vs model

An interpretation is a way to assign meaning to expressions of a language. One way to assign meaning is to take an expression in the object language (here TLA$^+$), and map it to an expression in the metalanguage (here ZF) [8, p.292]. A model is an interpretation in which all the theorems of a formal theory are interpreted as statements that are theorems of the metatheory.

We base our definition of interpretation for first-order TLA$^+$ on the literature. Vanzetto defines a translation of TLA$^+$ to input for SMT solvers [14]. Merz formalizes TLA$^+$ in the Isabelle theorem prover [15], for TLAPS (the TLA proof system), including functions [16]. Models of first-order TLA* are defined in [17, pp.80–86]. Lamport refers to Leisenring's study [18] of the operator CHOOSE. Leisenring defines models for ZFε[7] [18, p.18], but not for modal logic. A semantics for TLA$^+$ can be given by pairing a model as defined by Leisenring with a behavior σ, similarly to [14, 17, 19, 20]. Following Lamport, we use TLA$^+$ syntax for ZF itself [8, p.292], including function syntax. For example, the definition of behaviors as functions with domain Nat [8, p.315].

By interpretation of TLA$^+$ in ZF we mean an operator $I(\_, \_)$ and a set σ such that

- $IsABehavior(\sigma)$, and
- the mapping $I(\_, \_)$ defines the meaning $I(e, s)$ of each basic state function e in each state s. Let[8]
  $$s[\![e]\!]_I \triangleq I(e, s)$$
  Constant expressions have the same meaning in each state, so we can omit s and write $[\![e]\!]_I$.

The role of a behavior σ is to map each index $n \in Nat$ and variables[9] to values in each index $n \in Nat$. We will denote that a temporal formula F is interpreted as true in the metatheory as [17]

$$I, \sigma \models F$$

[7]ZF abbreviates Zermelo-Fraenkel set theory. ZFε denotes ZF extended with the operator CHOOSE. The operator CHOOSE is Hilbert's ε operator [18], [21, p.9]. We use ZFε, so we omit ε and write ZF.

[7]A *basic state function* is a particular form of TLA$^+$ expression. The TLA$^+$ syntax defines countably many expressions. The *meaning* of a basic state function is a mapping from *all* states, so an operator, but not a function [8, p.311].

[8]The expression $s[\![e]\!]_I$ extends the notation $s[\![e]\!]$ of [8, p.311]. A similar notation is $s[\![x]\!]^{I,\alpha}$ in [17, p.81].

The assignment of values to constants (rigid variables) is specified by I. We could define a separate operator Const_I, but we will use I for that purpose too, in the recursive interpretation defined later. The other observation is that we will writte $\mathit{Flatten}(\ldots, I)$ and recurse to $\mathit{ApplyFunc}_I$ when I is given to the operator *Flatten*, and to $\mathit{ApplyFunc}_R$ when R is given to *Flatten*.

A *model* of TLA$^+$ is an interpretation in which each theorem is interpreted as a statement provable in the metatheory. We use an operator $\mathit{IsAModel}(I(\_,\_), \sigma)$ to signify this, by writing the predicate $\mathit{IsAModel}(I, \sigma)$.

3.2.2 Recursively structured interpretation

Structured interpretations In the proofs, we assume for some interpretation I_A, σ that $\mathit{IsAModel}(I_A, \sigma)$, and modify the first-order interpretation operator I_A to I_B so that both:

1. the resulting interpretation I_B, σ has some desired properties[10], and
2. $\mathit{IsAModel}(I_B, \sigma)$.

It can be difficult to define I_B in terms of I_A so as to achieve both of these goals. An example of what can go wrong is given in Section 6.3. The difficulty lies in that I_A takes entire TLA$^+$ expressions as arguments. We have to change not only the interpretation of those symbols that we are primarily interested in, but also all other expressions that contain those symbols, to ensure that all axioms and proof rules are true in I_B too.

The standard way of defining a first-order interpretation I is inductively [8, Ch.16]. This is the case in the literature cited earlier. TLA$^+$ is a modal logic, so the interpretation I, σ of a temporal formula is defined in terms of actions over steps [8, p.315], of actions in terms of state predicates over states [8, p.313], and of constant expressions [8, pp.311]. The interpretation I is defined inductively using the operators:

- $\mathit{Tokenize}(e)$ for a TLA$^+$ expression e is a tuple of strings (tokens).
- $\mathit{Flatten}(\mathit{seq}, s, I(\_,\_))$ takes a sequence of tokens and interprets it in ZF.
- $\mathit{ApplyFunc}_I(u, v)$, an operator that takes the interpretation u of an expression e_1, the interpretation v of another expression e_2, and gives the interpretation of the expression "$e_1[e_2]$".
- $\mathit{MakeFunction}(S)$ for a set S of 2-tuples returns a function.
- $\mathit{DomainOf}(a)$ interprets function domain syntax.

[9] There are variables and constants in TLA$^+$. Sometimes, constants are called "rigid variables" [8, p.110], and the variables "flexible variables". A behavior σ assigns values only to flexible variables. The values of constants are defined by the first-order interpretation I.

[10] Modifying one model to obtain another is a method widely used for proving relative consistency results. The method of forcing [22] is one approach for constructing new models. We do not use forcing, because we modify the interpretation of functions without changing the domain of discourse.

We describe these operators using a combination of definitions and examples

$$\begin{aligned} \mathit{Tokenize}(\mathit{str}) \;\triangleq\;& \text{CHOOSE } \mathit{seq} \in \mathit{Seq}(\text{STRING}) : \\ & \land\ \mathit{seq} \in \mathit{TLAPlusGrammar.Module} \\ & \land\ \mathit{str} = \mathit{Concatenate}(\mathit{seq}) \end{aligned}$$

with *Concatenate* defined as we usually understand it, *Seq* from the standard *Sequences* module [8, Fig.18.1, p.341], and *TLAPlusGrammar* [8, p.278] [11]. Rigid quantification is defined in [8, pp.88, 109] as

$$\sigma \models (\forall\, r : F) \;\triangleq\; \forall\, r : (\sigma \models F)$$

The modification of constant symbols in the first-order interpretation is not mentioned, and the same symbol r occurs in object language (TLA$^+$) as in the metalanguage (ZF). We need to keep track of these details, so we base our definition on[12] [14, p.31]

$$\begin{aligned} & I, \sigma \models \forall\, r : \varphi \;\triangleq \\ & \qquad \forall\, u : \text{LET} \\ (4) \qquad & \qquad\quad I|_{r \triangleq u}(\mathit{seq}, s) \;\triangleq\; \text{IF } \mathit{seq} = \langle \text{"}r\text{"} \rangle \\ & \qquad\qquad\qquad\qquad\qquad\quad \text{THEN } u \\ & \qquad\qquad\qquad\qquad\qquad\quad \text{ELSE } I(\mathit{seq}, s) \\ & \qquad \text{IN} \quad I|_{r \triangleq u}, \sigma \models \varphi. \end{aligned}$$

Function constructors are interpreted as

$$\begin{aligned} & \mathit{Flatten}(\mathit{Tokenize}(\text{"}[x \in e_1 \mapsto e_2]\text{"}), s, I) = \\ & \mathit{MakeFunction}(\{ \\ & \quad \langle\, r,\ \mathit{Flatten}(\mathit{Tokenize}(\text{"}e_2\text{"}), s, I|_{x \triangleq r})\,\rangle : \\ & \qquad r \in \mathit{Flatten}(\mathit{Tokenize}(\text{"}e_1\text{"}), s, I) \\ & \}) \end{aligned}$$

So, a function is determined by the values of e_2 for x ranging over values in e_1, but not outside. Function domain syntax is interpreted as

$$\mathit{Flatten}(\mathit{Tokenize}(\text{"DOMAIN } e\text{"}), s, I) = \mathit{DomainOf}(\mathit{Flatten}(\mathit{Tokenize}(\text{"}e\text{"}), s, I))$$

Function application is interpreted as

$$\begin{aligned} \mathit{Flatten}(\mathit{Tokenize}(\text{"}e_1[e_2]\text{"}), s, I) = \mathit{ApplyFunc}_I\big(& \mathit{Flatten}(\mathit{Tokenize}(\text{"}e_1\text{"}), s, I), \\ & \mathit{Flatten}(\mathit{Tokenize}(\text{"}e_2\text{"}), s, I)\big). \end{aligned}$$

[11]If no such *seq* exists, then *Tokenize*(*str*) is some unknown value, but we don't care, because this happens only when, by definition of *TLAPlusGrammar*, *str* is *not* a syntactically correct TLA$^+$ module. Strictly speaking, not all syntactically acceptable module strings have meaning [8, Chapters 16, 17], but we can ignore those cases, because the result for those strings is irrelevant to our study of TLA$^+$.

[12]In particular, the definition of $\mathit{truth}_{\mathcal{M}}(\forall\, x : \varphi)$ there. There is a typo in the subscript, where $\mathcal{I}$ and $v \oplus (x \mapsto d)$ need to be swapped.

With the definition sketched above, we can reduce the meaning of any TLA$^+$ formula to the meaning of variables and constants [8, p.310] in ZF.

The tokenizer in action The operator *Tokenize* takes a string *str* (a finite sequence of characters), and chooses *seq*, a sequence of strings (each of them a token) that belongs to the TLA$^+$ grammar, so that concatenating the strings in *seq* yields *str*. For example,

$$\mathit{Tokenize}(\text{“}p \in q\text{”}) = \langle \text{“}p\text{”}, \text{“} \in \text{”}, \text{“}q\text{”} \rangle.$$

The string “$p \in q$” has been split into the sequence $\langle \text{“}p\text{”}, \text{“} \in \text{”}, \text{“}q\text{”} \rangle$, where each of “$p$”, “ $\in$ ”, “q” is a string. This agrees with our perception of what a tokenizer program is doing.

The operator *Flatten* interprets the tokenizer's result in state s as follows

$$\begin{aligned} \mathit{Flatten}(\mathit{Tokenize}(\text{“}p \in q\text{”}, s, I) &= \mathit{Flatten}(\langle \text{“}p\text{”}, \text{“} \in \text{”}, \text{“}q\text{”} \rangle, s, I) \\ &= \mathit{Flatten}(\langle \text{“}p\text{”} \rangle, s, I) \ \in \ \mathit{Flatten}(\langle \text{“}q\text{”} \rangle, s, I) \end{aligned}$$

Is there such an interpretation? We outlined an inductive definition of interpretation. The operator *Flatten* can be defined in more detail as in [14]. We are not going to give an exhaustive definition, neither prove relative consistency of TLA$^+$ with respect to ZF. The parser and translator code in TLAPS and TLC[13] is evidence that it is possible to define these operators.

By Gödel's second incompleteness theorem, if TLA$^+$ is consistent, then TLA$^+$ cannot prove it is consistent. By Gödel's completeness theorem, if TLA$^+$ is consistent, then it has a model with a set as domain of discourse [23, Sec.5.2].

We are going to assume that a relative interpretation [24, Def. I.16.11, p.99] of TLA$^+$ in ZF is available, and modify it to obtain another relative interpretation that satisfies a desired property. Assuming that the starting relative interpretation proves relative consistency [24, Cor. I.16.14, p.101], relative consistency follows for the second interpretation. We need to refer to behaviors, so we cannot work directly within TLA$^+$ to prove relative consistency by interpreting the second TLA$^+$ theory in terms of the starting one. Instead, our proof could be viewed as a relative interpretation of the ZF underlying the second TLA$^+$ theory within the ZF underlying the starting TLA$^+$ theory.

Note that relative interpretation via relativization relies on class-sized models, and care is then required that the interpretation be defined in the metatheory, due to Tarski's theorem on the undefinability of truth [24, p.84].

Another way to view the proofs is in terms of set-sized models. Assuming that a model with set as domain of discourse exists, we construct another model with the same domain, but different interpretation for functions. The assumption that a set-sized model exists is equivalent to assuming that ZF is consistent [23, Sec.5.2].

[13] TLC is a model checker for TLA$^+$.

3.2.3 TLA$^+$ function axioms

We summarize the definitions and axiom schemata of TLA$^+$ that describe functions. The symbol e stands for any TLA$^+$ expression.

$$IsAFunction(f) \triangleq f = [x \in \text{DOMAIN } f \mapsto f[x]]$$

$$\forall f, g : \left(\begin{array}{l} \land IsAFunction(f) \\ \land IsAFunction(g) \end{array} \right) \Rightarrow (f = g) \equiv \begin{array}{l} \land \text{DOMAIN } f = \text{DOMAIN } g \\ \land \forall x \in \text{DOMAIN } f : f[x] = g[x] \end{array}$$

$$(\text{DOMAIN } [x \in S \mapsto e]) = S$$

$$\forall y \in S : [x \in S \mapsto e][y] = \text{LET } x \triangleq y \text{IN } e$$

$$IsAFunction([x \in S \mapsto e])$$

$$\forall f : (f \in [S \to T]) \equiv \begin{array}{l} \land IsAFunction(f) \\ \land S = \text{DOMAIN } f \\ \land \forall x \in S : f[x] \in T \end{array}$$

$$[f \text{ EXCEPT } ![d] = e] \triangleq [y \in \text{DOMAIN } f \mapsto \text{IF } y = d \text{ THEN } \begin{array}{l} \text{LET } @ \triangleq f[d] \\ \text{IN } e \end{array} \text{ ELSE } f[y]]$$

This summary is given only for convenience. For details, consult [8, 25, 16].

It should be emphasized that TLA$^+$ functions are interpreted "shallowly" by the operators *ApplyFunction* and *MakeFunction*. These define TLA$^+$ functions separately from ZF functions. We could have followed a "deeper" approach, but it would have only made the proofs less readable.

3.2.4 When realizability is unprovable

Modifying the value of constant symbols does not affect the truthness of axioms and proof rules in a model of TLA$^+$ that is an interpretation with the recursive structure outlined above. Care is needed when modifying the operator *ApplyFunc* to ensure that function axioms be preserved. We use the following claim, whose proof is sketched, but omitted.

Proposition 5 (Independence from function application outside domain)

ASSUME: *1.* $IsAModel(J, \sigma)$

2. J has the recursive structure outlined above.

3. R is an interpretation obtained from J by replacing the operator $ApplyFunc_J$ *with an operator* $ApplyFunc_R$

4. $\forall u, v : \lor v \notin DomainOf(u)$
$\qquad\qquad \lor ApplyFunc_R(u, v) = ApplyFunc_J(u, v)$

PROVE: $IsAModel(R, \sigma)$

PROOF SKETCH: on page 36.

Another way to describe the above change is based on how TLAPS translates TLA$^+$ to SMT, which can be written as

$$\mathit{ApplyFunc}(u, v) = \textsc{if}\ v \in \mathit{DomainOf}(u)\ \textsc{then}\ a(u, v)\ \textsc{else}\ \omega(u, v),$$

where the operator ω remains uninterpreted [14, Eq.(4.12), §4.4.2, p.78]. What we do amounts to modifying only ω.

Lemma 6 (Unprovability of realizability)
ASSUME: *1. TLA$^+$ is sound.*
2. ZF NEW $J(\_,\_), \sigma$
3. J, σ is an interpretation with the recursive structure described earlier.
4. $\mathit{IsAModel}(J, \sigma)$
5. TEMPORAL $\varphi(x, h)$
6. The symbols f, m do not occur in the expression $\varphi(x, h)$.
7. No[14] function syntax occurs in the expression $\varphi(x, h)$ (function application, constructors, and related constructs).
8. $\psi(x, h, f) \ \triangleq\ \boldsymbol{\exists}\, m : \Box[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}$
9. $\mathit{CanLoseOutside} \ \triangleq\ \forall\, S : \boldsymbol{\exists}\, x, h : \land \neg\varphi(x, h)$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \land \Box(\langle x, h\rangle \notin S)$
10. $\models \mathit{CanLoseOutside}$
11. $G \ \triangleq\ \exists f : \land \mathit{IsAFunction}(f)$
$\qquad\qquad\qquad \land \boldsymbol{\forall}\, x, h : \psi(x, h, f) \Rightarrow \varphi(x, h).$

PROVE: $\not\vdash G$

PROOF: on page 37.

In words, prove that (arbitrary[14] memory) realizability of φ cannot be proved, unless, for any behavior that violates φ, eventually $\langle x, h\rangle \in S$.

In other words, outside its domain, f plays arbitrarily. The behavior satisfies ψ, but may violate φ, and we cannot prove that it doesn't. We have shown that f should matter only for behaviors that eventually enter some set S, $\Diamond(\langle x, h\rangle \in S)$. It follows that any provably realizable specification φ should satisfy

$$\neg(\forall\, S : \boldsymbol{\exists}\, x, h : \neg\varphi(x, h) \land \Box(\langle x, h\rangle \notin S) \equiv$$
$$\exists\, S : \boldsymbol{\forall}\, x, h : \Box(\langle x, h\rangle \notin S) \Rightarrow \varphi(x, h)$$

The above case is one alternative. The other alternative is

$$\neg\Box(\langle x, h\rangle \notin S) \equiv \Diamond(\langle x, h\rangle \in S) \equiv \begin{array}{l} \lor\ \Box(\langle x, h\rangle \in S) \\ \lor\ \land\ \Diamond(\langle x, h\rangle \in S) \\ \quad\ \land\ \Diamond(\langle x, h\rangle \notin S) \end{array}$$

[13]This assumption can be relaxed to: under the moderate interpretation of Boolean operators, the meaning of φ is independent of function values that correspond to arguments outside their domain. However, this will complicate the proof, without adding to our purpose.

[14]This is a strong statement. Neither finiteness, nor even type invariance of m is required.

For specifications that contain enough type invariants, there is some set S such that a behavior $\sigma \models \Diamond(\langle x, h\rangle \notin S)$ has a finite prefix that either satisfies or violates the specification (depending on whether the environment or system violated their type invariant first).

For the same reasons as above, we shouldn't expect any function to be provably winning depending on the initial prefix of a behavior (behavior states before the earliest state that satisfies $\langle x, h\rangle \in S$). So, we will regard the previous result as support that if φ is provably realizable, then behaviors that start from outside S should satisfy φ, independently of f. In other words, outside S, the initial condition should determine whether a behavior is winning or losing. Formally,

(5)

$$\models \exists S : \forall u, r \notin S : \exists w \in \text{BOOLEAN} : \boldsymbol{\forall} x, h : (\langle x, h\rangle = \langle u, r\rangle) \Rightarrow (w = \varphi).$$

So, DOMAIN $f \subseteq S$ suffices to consider all candidate winning strategies.

For example, if

$$\varphi \triangleq (y \in S_y \land \dots) \overset{+}{-\!\!\triangleright} (x \in S_x \land \dots),$$

then

- $(x \notin S_x) \Rightarrow \neg\varphi$, and
- $(x \in S_x) \land (y \notin S_y) \Rightarrow \varphi$,

so for $(x \notin S_x) \lor (y \notin S_y) \equiv \langle x, y\rangle \notin S_x \times S_y$, strategy f does not matter. This partitioning of initial conditions is shown schematically in Fig. 2. These are all the possible cases, but not all should occur. Clearly, if $x \in S_x$ appears in (or is implied by) the guarantee in $\overset{+}{-\!\!\triangleright}$, the specification is unrealizable. This case is analyzed further in Section 5.2.4.

Notice that $x \notin S_x$ is not a set of values for x, and $(x \in S_x) \land (y \notin S_y)$ not a set of values for y [8, p.66], [6, 7]. No function f can be designed to play in a known way from so many initial conditions, because DOMAIN f would have to be too large to be a set. Nevertheless, we need only reason about f in those behaviors that satisfy $(x \in S_x) \land (y \in S_y)$. Proving that f does not matter from other initial conditions allows us to ignore those initial conditions when designing f. Such a proof ensures that it suffices for DOMAIN f to contain all initial conditions from where f can affect the outcome, and aids in mechanization of finding f (the other aid is the presence of type invariants, for similar reasons).

Focusing on a set of initial conditions is independent of whether any initial states violate φ. By the above, we showed that an initial condition in the form of set containment should appear as an assumption (or within the realization ψ). This is necessary for any TLA$^+$ definition of synthesis, due to our unprovability result.

It also aids mechanization, but may not be enough for mechanization. Type invariants are an extra aid for mechanization, which may be unnecessary for certain types of φ, assuming suitable theorem-proving capabilities.

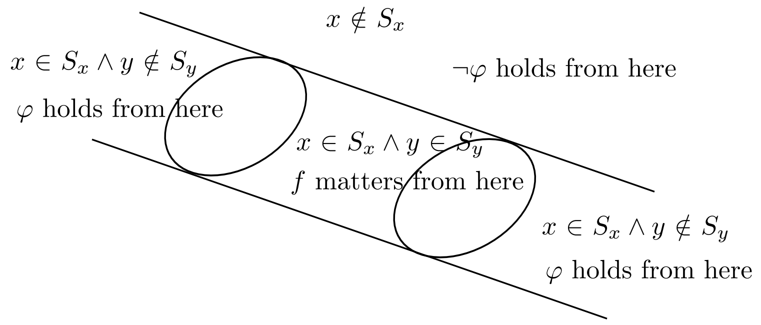
$x \notin S_x$
$x \in S_x \land y \notin S_y$
$\neg\varphi$ holds from here
φ holds from here
$x \in S_x \land y \in S_y$
f matters from here
$x \in S_x \land y \notin S_y$
φ holds from here

Figure 2: Initial conditions from where the strategy f matters. The collection of values for x that satisfy the predicate $x \notin S_x$ is too large to be a set. Likewise for $y \notin S_y$. Unlike them, the values for $\langle x, y\rangle$ that satisfy the predicate $x \in S_x \land y \in S_y$ *does* form a set. This allows for a function f to be *designed* for acting within a set as its domain.

3.3 Placing initial conditions

If some initial conditions violate φ, i.e.,

$$\exists\, u, r \,:\, (\langle x, h\rangle = \langle u, r\rangle) \Rightarrow \neg\varphi(x, h)$$

then we must include an initial condition I in the realization ψ, for example

$$I(x, h) \land \Box[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}$$

and select I so that $\models I(x, h) \Rightarrow \mathit{Init}(x, h)$ (Lemma 4). We decide to not include an initial condition in ψ, to keep it simpler. So, no initial state should violate φ. (Otherwise, no matter how we play from a violating state, we have already lost.)

Still, we must include an initial condition in the assumption of φ, because otherwise no function can be *proved* to implement φ (Section 3.2).

We will see in Section 5.2.4 that the definition[15] of $\overset{+}{\rightharpoonup}\!\!\triangleright$, together with the untyped nature or TLA$^+$, require including a type precondition on environment variables in the guarantee (or defining a slight variant of $\overset{+}{\rightharpoonup}\!\!\triangleright$). This is an additional reason for not adding an initial condition to ψ, because then the initial condition would appear at two places (Of course, this doesn't apply to the aforementioned variant of $\overset{+}{\rightharpoonup}\!\!\triangleright$.)

Assume that we use the operator $\overset{+}{\rightharpoonup}\!\!\triangleright$ to define $\varphi \;\triangleq\; A \overset{+}{\rightharpoonup}\!\!\triangleright G$. By Proposition 9, φ is vacuous if A is unsatisfiable. Assume $\exists\, x, h \,:\, A(x, h)$ (satisfiable assumption). By Proposition 12, initial states that violate Init_G violate φ. So, it must be $\mathit{Init}_G \;\triangleq\; \text{TRUE}$. The only remaining location to place the initial condition assumption is Init_A.

Advantages of writing the synthesis problem and the specification φ in this way are:

- A synthesis problem is defined by the controllability predicates μ_{ik}, and a single property φ, instead of two separate properties I and φ. This is simpler. It also collects everything about temporal behavior within one property, φ.

- The initial condition for every variable appears in the assumption A, emphasizing that it is the environment's responsibility, not the component's. In other words, *we* should start the component f from admissible initial conditions, which is what we mean by including I in ψ. Putting I inside A, instead of inside the realization ψ, groups it with other assumptions, all assumptions at one place.

Disadvantages:

- Symmetry is lost. The property $\varphi_{env} \quad \triangleq \quad G \overset{+}{\rhd} A$ does not anymore express a specification for constructing an environment component, unless we reverted to defining synthesis for the environment using an initial condition I inside ψ. What really happened is that, by using I, the asymmetry was factored out as I, so asymmetry was shifted outside φ. This alowed φ_{env} to be obtained by simply swapping G and A. In contrast, with the property
$$\varphi_{sys} \;\triangleq\; (I \wedge A) \overset{+}{\rhd} G,$$
where no initial states violate G (for the reasons explained above), the environment should be specified with the property
$$\varphi_{env} \;\triangleq\; (I \wedge G) \overset{+}{\rhd} A,$$
where no initial states violate A (for the same reasons).

- We cannot synthesize strategies for closed-system properties. So, $\varphi \quad \triangleq \quad G$ is unrealizable. We must either write it as $I \overset{+}{\rhd} G$, or place an initial condition I within the realization ψ.

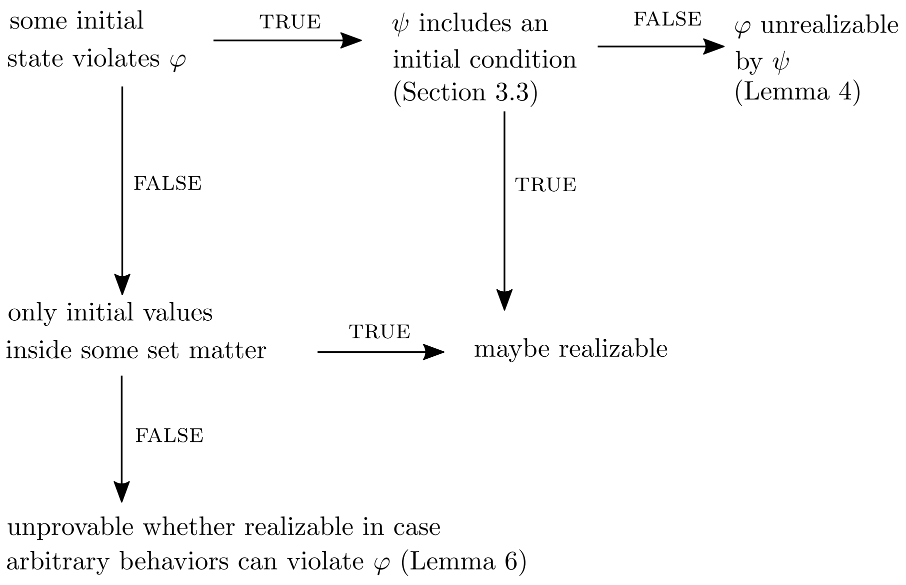
some initial state violates φ
TRUE
ψ includes an initial condition (Section 3.3)
FALSE
φ unrealizable by ψ (Lemma 4)
FALSE
TRUE
only initial values inside some set matter
TRUE
maybe realizable
FALSE
unprovable whether realizable in case arbitrary behaviors can violate φ (Lemma 6)

Figure 3: Summary of results about initial conditions.

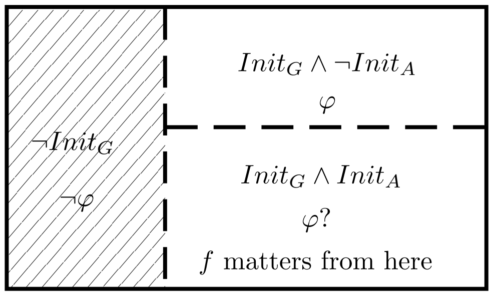

Figure 4: Role of initial conditions in an open-system property of the form $(Init_A \wedge A) \overset{+}{-\!\!\rhd} (Init_G \wedge G)$.

4 Defining realizability and synthesis

4.1 Sharing variables

We follow an approach noted by Lamport [1], along the lines of work by Abadi and Lamport [10]. This approach formulates the synthesis problem using, for each variable x, an action μ_x. If a step satisfies the action μ_x, then it is called a μ_x-step. Any change of variable x in a μ_x-step is attributed to player f_x, possibly among other players.

This action promises to f_x that each step either stutters x, or is a $\neg\mu_x$-step, or, if f_x so wants, then a μ_x-step that changes x will eventually occur. In any μ_x-step that changes x, function f_x controls x'.

The action μ_x tells the system that it can affect certain variables in certain steps. Conversely, it lets environment variables unconstrained by system strategies. Wild environment behavior is then to be rejected by the assumption in φ. The definition of action μ_x should be part of the open-system specification module that defines the system property φ.

4.2 Realizability

Realizability is a decision problem, and synthesis the associated search problem. We can define realizability in TLA$^+$ as in Fig. 5, based on Lamport's definition [1] (see also Appendix C). This definition motivates the definition of Fig. 6, which is in canonical form and more general. The requirements for finite domains ensure that f, g can be implemented in the real world.

With the definition of Fig. 5, a given temporal formula φ is *realizable* with the action μ if we can prove the

$$\text{THEOREM} \quad \mathit{Realizability}(\varphi, \mu)!\mathit{IsRealizable}.$$

Note the action's meaning

$$\begin{array}{l} \vee \neg\mu \\ \vee\; x' = f[v] \wedge m' = g[v] \\ \vee\; \langle m, x, y\rangle' = \langle m, x, y\rangle \end{array}$$

This formula says that

- any change of x, m in a $\neg\mu$-step is attributed to the environment (disjunct $\neg\mu$),
- changes of x and m in a μ-step are attributed to f and g (disjunct $x' = f[v] \wedge m' = g[v]$), or
- a stuttering step occurs (disjunct $\langle m, x, y\rangle' = \langle m, x, y\rangle$).

The brevity of this definition owes to the definition of a function and the untypeness of TLA$^+$. Variants are possible, by writing two safety and liveness

```
---------------------------- MODULE RealizabilityFirst ----------------------------
EXTENDS FiniteSets
STATE $\mu(\_,\_)$
TEMPORAL $\varphi(\_,\_)$
$IsAFunction(f) \triangleq f = [u \in \text{DOMAIN } f \mapsto f[u]]$
  ------------------------------ MODULE Inner ------------------------------
  VARIABLES $x, y$
  CONSTANTS $f, g, m_0$
  $OnlyAllowedChanges \triangleq$
    $\forall r : \forall \langle m, p, q \rangle \in \text{DOMAIN } f \cup \text{DOMAIN } g :$
    LET $v = \langle m, p, q \rangle$
    IN  $\lor \langle f[v], g[v] \rangle = \langle p, q \rangle$
        $\lor \mu(v, \langle r, f[v], g[v] \rangle)$
  $Realization(m) \triangleq$
    LET
      $v \triangleq \langle m, x, y \rangle$
      $A \triangleq \land x' = f[v]$
           $\land m' = g[v]$
    IN  $\land m = m_0$
        $\land \Box[\mu(v, v') \Rightarrow A]_v$
        $\land \Box\Diamond \lor \langle \mu(v, v') \rangle_v$
              $\lor \land x = f[v]$
                $\land m = g[v]$
  $Realize \triangleq \land IsAFunction(f) \land IsFiniteSet(\text{DOMAIN } f)$
            $\land IsAFunction(g) \land IsFiniteSet(\text{DOMAIN } g)$
            $\land OnlyAllowedChanges$
            $\land (\exists m : Realization(m)) \Rightarrow \varphi(x, y)$
  -------------------------------------------------------------------------
$Inner(f, g, m_0, x, y) \triangleq$ INSTANCE Inner
$IsRealization(f, g, m_0) \triangleq \forall x, y : Inner(f, g, m_0, x, y)!Realize$
$IsRealizable \triangleq \exists f, g, m_0 : IsRealization(f, g, m_0)$
-----------------------------------------------------------------------------------
```

Figure 5: A module that defines realizability by constraining the strategy f to request only $\langle m, x \rangle$-nonstuttering steps. The module *FiniteSets* can be found in [8, Fig.18.2, p.341].

```
---------------- MODULE Realizability ----------------
EXTENDS FiniteSets
STATE $\mu(\_, \_)$
TEMPORAL $\varphi(\_, \_)$
$IsAFunction(f) \triangleq f = [u \in \text{DOMAIN } f \mapsto f[u]]$
    ---------------- MODULE Inner ----------------
    VARIABLES $x, y$
    CONSTANTS $f, g, m_0$
    $Realization(m) \triangleq$
      LET
        $v \triangleq \langle m, x, y \rangle$
        $A \triangleq \land x' = f[v]$
             $\land m' = g[v]$
      IN $\land m = m_0$
         $\land \Box[\mu(v, v') \Rightarrow A]_v$
         $\land \mathrm{WF}_{\langle m, x \rangle}(\mu(v, v') \land A)$
    $Realize \triangleq \land IsAFunction(f) \land IsFiniteSet(\text{DOMAIN } f)$
              $\land IsAFunction(g) \land IsFiniteSet(\text{DOMAIN } g)$
              $\land \big(\boldsymbol{\exists}\, m : Realization(m)\big) \Rightarrow \varphi(x, y)$
    ----------------------------------------------
$Inner(f, g, m_0, x, y) \triangleq$ INSTANCE Inner
$IsRealization(f, g, m_0) \triangleq \boldsymbol{\forall}\, x, y : Inner(f, g, m_0, x, y)!Realize$
$IsRealizable \triangleq \exists f, g, m_0 : IsRealization(f, g, m_0)$
------------------------------------------------------
```

Figure 6: A module that defines realizability.

pairs, one for variable x, and another for m. The memory can be shifted to the property φ itself, in which case m would be mentioned within φ.

The liveness condition requires weakly fair scheduling

$$\begin{array}{l} \Box\Diamond \lor \langle\mu\rangle_v \\ \quad\ \lor \land\, x = f[v] \\ \qquad\ \land\, m = g[v] \end{array} \quad \equiv \quad \left(\begin{array}{l} \Diamond\Box \lor x \neq f[v] \\ \qquad\ \lor\, m \neq g[v] \end{array} \right) \Rightarrow \Box\Diamond\langle\mu\rangle_v .$$

In words, if f or g continuously request to make a change, then eventually $\mu \land (v' \neq v)$. By the safety conjunct, this implies eventually

$$\langle x', m' \rangle = \langle f[v], g[v] \rangle .$$

Specifically for games with full information[16](where $\mu = \neg\text{UNCHANGED } \langle m, x, y\rangle$), the negated fairness condition means that f, g want to change x, m, but a change never happens

$$\begin{array}{l} \neg\Box\Diamond \lor \langle\text{TRUE}\rangle_v \\ \qquad\ \lor \land\, x = f[v] \\ \qquad\quad \land\, m = g[v] \end{array} \quad \equiv \quad \begin{array}{ll} \Diamond\Box \land v = v' & \text{nothing changes} \\ \qquad \land\, \langle x, m\rangle \neq \langle f[v], g[v]\rangle & \text{request for change} \end{array}$$

Comments on the definition

- Why not write $\langle x, m\rangle' = f[v]$, instead of using two functions f and g? Because, a function f can falsify $\langle x, m\rangle' = f[v]$ by simply taking values that are tuples of different length. This leads to trivial realizability, as proved in Proposition 15. Clearly, when we think of "implementation", we do not mean this case.

  A more complicated and less readable alternative is to add the constraint

  $$\forall\, v \in \text{DOMAIN } f \,:\, \mathit{IsATuple}(f[v]) \land \mathit{Len}(f[v]) = 2.$$

- Universal temporal quantification of v ($\boldsymbol{\forall}\, v$) cannot be avoided, because then, by the linear nesting of quantifiers, for each behavior σ, we could pick a different function f.
- Existential temporal quantification of m ($\boldsymbol{\exists}\, m$) cannot be avoided, because variable m is an internal variable of the implementation, so if it occurs in φ, then it should not be constrained by a realization, except via x.
- To define controllability, we need the action μ.
- To avoid unprovability of realizability we have to include a set containment as initial condition in φ (Lemma 6).
- To quantify over a mapping f, it must be a set, so a function (Section 2).

[16]Asynchronous reduces to stutter invariance within the assumption's action.

- Instead of adding an (environment) assumption (H in Appendix C) about infinite memory h, we find it simpler and explicit to design the memory update function g. This also makes the implementation self-contained.

- The liveness condition about μ cannot be placed in the assumption of φ, because the liveness condition for f cannot be a conjunct of the implementation (that would allow any f that never stutters to be a winning strategy). So, the liveness conjunct should mention both f and μ. But, the implementation f should not occur as a constant in φ.

There is no way to "prove" these observations right or wrong, because what we are doing is *specifying* what synthesis means. No mechanized way can confirm that this definition matches what we think. The best we can hope for is to avoid errors, by constructing simple counterexamples [8, p.76].

4.3 Synthesis

Synthesis corresponds to the proof step

$$\textsc{pick}\ f, g, m_0\ :\ \mathit{Realizability}(\varphi, \mu)!\mathit{IsRealization}(f, g, m_0).$$

By the definition of PICK [26, Sec.7.4.5, p.25] the above expands to the proof steps

$\langle 1 \rangle 1.$ $\mathit{Realizability}(\varphi, \mu)!\mathit{IsRealizable}$
$\langle 1 \rangle 2.$ SUFFICES: ASSUME: 1. NEW f, g, m_0
2. $\mathit{Realizability}(\varphi, \mu)!\mathit{IsRealization}(f, g, m_0)$
PROVE: Q.E.D.

In principle, one could require *IsRealizable* to hold, and then CHOOSE f, g, m_0. However, CHOOSE for temporal formulae is intentionally omitted from the current definition of TLA$^+$ [8, p.110].

5 Who changed each variable

5.1 Behaviors

5.1.1 Interleaving

A behavior is interleaving if each nonstuttering[17] step is attributed to exactly one player [8, p.137]. A step is attributed to a player by using an action[18] μ_i.
With this definition of attribution, a behavior is interleaving if it satisfies the (mutual exclusion) property[19]

$$(6)\qquad \begin{aligned} C &\triangleq 0..n \quad \text{[component indices]} \\ \mathit{Interleaving} &\triangleq \Box[\forall j \in C : \forall i \in C \setminus \{j\} : \mu_j \Rightarrow \neg \mu_i]_v \end{aligned}$$

This definition works for both disjoint-state and shared-state specifications. In words, if each nonstuttering step is attributed to at most one player, then the behavior is interleaving. If any step is attributed to more than one players, then the behavior is noninterleaving. In other words, a noninterleaving behavior is one that is not interleaving. Interleaving refers only to variables mentioned in the specification[20].

Also, if μ_i allows stuttering steps, then any stuttering will be considered as interleaving. For example, $\mu_i \triangleq \text{TRUE}$ is problematic. This μ_i attributes *all* steps to a player, including stuttering steps. As a result, this leaves indefinite stuttering as the only[21] behaviors that satisfy the property *Interleaving*. To avoid this, we can either require that $\mu_{ik} \Rightarrow (x_k \neq x'_k)$, or, within the predicate *Interleaving*, conjoin $x_k \neq x'_k$ to each μ_{ik}, to obtain $\mu_{ik} \wedge (x_k \neq x'_k)$. The problem with the latter is that it leads to a mismatch between what attribution of steps categorizes a behavior as interleaving, and what attribution we obtain by using μ_{ik} directly. In other words, attributing steps with μ_{ik} could yield a different result in this case. For this reason[22], we[23] require that μ_{ik} allow no

[17]If a variable does not change in a step, then there is no change to attribute to anyone. Otherwise, we need to choose who to attribute the change to.

[18]μ are actions. A Moore strategy can rely on requesting changes only at states that satisfy μ independently of next state.

[19]μ_i and x_k are metasyntactic notation.

[20]No behavior is *really* interleaving: A behavior assigns values to all variables. All the specifications that we write leave some variables unmentioned. So, for any behavior that is interleaving with the above definition, we can construct another behavior that differs only in how unmentioned variables change. We can choose to make unmentioned variables change in all steps. If we decide to attribute changes of unmentioned variables to the environment, then such a behavior shouldn't be considered interleaving. But we shouldn't characterize a behavior by thinking about unmentioned variables, because they don't matter to the problem at hand (otherwise they should have been mentioned). In other words, "interleaving" is not a genuine property of a behavior. It is a property defined relative to the variables mentioned in a given problem. If we restrict attention only to variables that occur in φ, then a specification can be called interleaving or not.

[21]For this example, $\Box[\forall j \in C : \forall i \in C \setminus \{j\} : \text{TRUE} \Rightarrow \text{FALSE}]_v \equiv \Box[\text{FALSE}]_v \equiv \Box(v' = v)$ (the last expression is not in TLA$^+$).

[22]Whether μ_{ik} allows stuttering steps or not has no effect on a realization, because $\Box[(\mu_{ik} \wedge (x'_k \neq x_k)) \Rightarrow (x'_k = f \ldots)]_{\langle \ldots x_k \ldots \rangle} \equiv \Box[\neg \mu_{ik} \vee (x'_k = x_k) \vee (x'_k = f \ldots)]_{\langle \ldots x_k \ldots \rangle} \equiv \Box[\mu_{ik} \Rightarrow (x'_k = f \ldots)]_{\langle \ldots x_k \ldots \rangle}$.

stuttering steps

$$\begin{aligned}(\models \mu_{ik} \Rightarrow (x'_k \neq x_k)) &\quad \equiv \\ (\models \langle \mu_{ik} \rangle_{x_k} \equiv \mu_{ik})&\end{aligned}$$

A special case is the following (orthogonality) definition given by Lamport [8, p.139]

$$\begin{aligned}&\Box[\exists\, k \in C : \forall\, i \in C \setminus \{k\} : v'_i = v_i]_v \equiv \\ &\Box[\forall\, j \in C : \forall\, i \in C \setminus \{j\} : (v'_j \neq v_j) \Rightarrow (v'_i = v_i)]_v \equiv \\ &\Box[\forall\, j \in C : \forall\, i \in C \setminus \{j\} : (v'_j = v_j) \lor (v'_i = v_i)]_v\end{aligned}$$

This definition works only for disjoint-state specifications. In other words, in order to apply this definition, first we need to partition the state into variable tuples $v_i \triangleq [k \in 0..n_i \mapsto \{x_j : j \in C\}]$. (Lamport's definition has $v'_i = v_i$, but stutter-invariance implies that the two formulae describe the same behaviors.) Algebraic manipulation shows that this is another way to write the disjointness definition given by Abadi and Lamport [27, p.514]

$$\begin{aligned}&\bigwedge{}_{i \neq j} \Box[(v'_i = v_i) \lor (v'_j = v_j)]_{\langle v_i, v_j \rangle} \equiv \\ &\forall\, j \in C : \forall\, i \in C \setminus \{j\} : \Box[(v'_i = v_i) \lor (v'_j = v_j)]_{\langle v_i, v_j \rangle} \equiv \\ &\forall\, j \in C : \forall\, i \in C \setminus \{j\} : \Box[(v'_i = v_i) \lor (v'_j = v_j)]_v\end{aligned}$$

An interesting discussion of the interleaving and non-interleaving specification styles, with examples, can be found in [28].

5.1.2 Shared or disjoint state

The actions μ_i attribute steps to players, not changes for each variable separately. If, for each variable (part) in an interleaving behavior, there exists a player, such that all steps that change that variable are attributed to that player, then state is disjoint [8, p.144]. If nonstuttering steps that change a variable in an interleaving behavior are attributed to more than one player, then state is shared.

These definitions work for interleaving behaviors only. They are too coarse for noninterleaving behaviors. The problem is that μ_i attribute each step to a player, but not changes of specific variables. A noninterleaving behavior contains a step attributed by μ_i, μ_j to players $i \neq j$. Which variables did each player change? If we attribute to each player the changes of all variables, then there are variables that both players change in the same step, so any noninterleaving behavior would have shared-state.

For this reason, we need to refine the attribution to players, from steps to changes of variables x_k. Replace the μ_i actions with the actions μ_{ik}. If a step satisfies action μ_{ik}, then we attribute to player i the change of variable x_k. Let $V \subseteq \mathit{Nat}$ be variable x_k indices and $C \subseteq \mathit{Nat}$ be component indices.

[23]Lamport requires that μ_i do not allow stuttering steps.

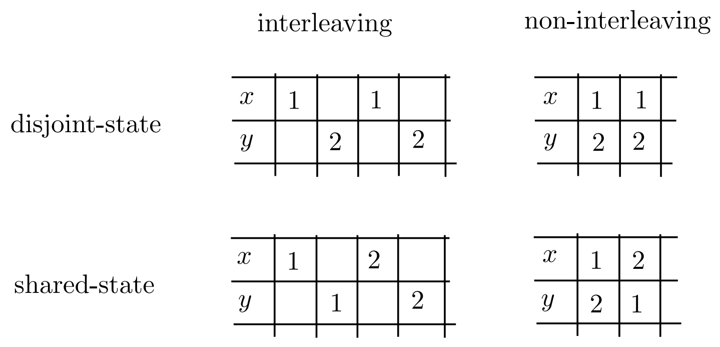

Figure 7: Schematic depiction of behaviors where changes to the variables x and y are attributed to players 1 and 2. If all changes to variable x are attributed to player 1, and all changes to variable y are attributed to player 2, then the behavior is disjoint-state. Otherwise, the behavior is shared-state. If, in each step, all changes to variables are attributed to at most one player, then the behavior is interleaving. ("at most" vs "exactly" depends on whether the μ_{ij} form a partition of unity).

With this refined definition, let

$$\mu_i \triangleq \exists\, k \in V : \mu_{ik} \tag{7}$$

so that interleaving still be defined as in Eq. (6). By substitution of Eq. (7) in Eq. (6), we obtain

$$\begin{aligned}
&\land\ \mathit{Interleaving} && \Rightarrow \\
&\land\ \forall\, i \in C : \mu_i = \exists\, k \in V : \mu_{ik} \\
&\quad \Box[\forall\, j \in C : \forall\, i \in C \setminus \{j\} : (\exists\, k \in V : \mu_{jk}) \Rightarrow \neg \exists\, k \in V : \mu_{ik}]_v \equiv \\
&\quad \Box[\forall\, j \in C : \forall\, i \in C \setminus \{j\} : (\exists\, k \in V : \mu_{jk}) \Rightarrow \forall\, k \in V : \neg \mu_{ik}]_v
\end{aligned} \tag{8}$$

Define as disjoint-state wrt $\{\mu_{jk}\}_{j\ \in\ C, k\ \in\ V}$ a behavior that satisfies the property

$$\mathit{DisjointState} \triangleq \forall\, k \in V : \exists\, i \in C : \forall\, j \in C \setminus \{i\} : \Box[\neg \mu_{jk}]_{x_k}.$$

In words, for each variable x_k, there exists a player i that can be considered as owner of x_k, because no change to x_k is attributed to any other player j.

5.1.3 Remarks

When by "specification" we mean "property", then the same definition applies to specifications too. But, when by "specification" we mean a TLA$^+$ module, then we should choose a specific property to apply the above definitions.

Variables from different components can stutter in the same steps, so we cannot use stuttering to attribute dynamics to players.

5.2 Properties

5.2.1 Interleaving

We defined what interleaving means for a behavior, given an attribution of changes of variable values to players.

Assume that actions μ_{ik} are defined. A property φ is interleaving with respect to $\{\mu_{ik}\}_{i,k}$ iff it describes only interleaving behaviors

$$\models \varphi \Rightarrow \mathit{Interleaving},$$

5.2.2 Shared or disjoint state

A property has disjoint-state with respect to $\{\mu_{ik}\}_{i,k}$ iff

$$\models \varphi \Rightarrow \mathit{DisjointState}.$$

A property φ that is not interleaving is called noninterleaving. A property φ without disjoint-state is shared-state.

5.2.3 Open-system specifications

Any realizable property of the form $A \overset{+}{\rhd} G$ that can be satisfied from any initial condition (i.e., strongest implied initial predicate is TRUE) allows a noninterleaving behavior, if in all states not all mentioned variables are controlled by one player only $(\forall\, x_0, \ldots, x_k : \exists\, i, j : \exists\, k, r : (i \neq j) \wedge (k \neq r) \wedge \mu_{ik} \wedge \mu_{jr})$.

PROOF: In Section 3.2 we showed that for winning to be provable in practice, the strategy f should matter only from some set of initial values. Suppose that the strongest initial predicate implied by φ is TRUE. Ensuring this initial predicate, but also ensuring that f matters only from a set of initial values, implies that $\mathit{Init}_A \not\equiv \text{TRUE} \Rightarrow \neg\mathit{Init}_A \not\equiv \text{FALSE}$. That φ is satisfiable from any initial state implies that $\mathit{Init}_G \equiv \text{TRUE}$.
Pick k, r and a behavior σ such that

$$\sigma \models \begin{array}{l} \wedge\ \mathit{Init}_G \wedge \neg\mathit{Init}_A \\ \wedge\ \mu_{ik} \wedge \mu_{jr} \end{array} \wedge \begin{array}{l} \wedge\ x_k \neq x'_k \\ \wedge\ x_r \neq x'_r \end{array}$$

By definition of $A \overset{+}{\rhd} G$, it is

$$(\mathit{Init}_G \wedge \neg\mathit{Init}_A) \Rightarrow (A \overset{+}{\rhd} G).$$

So, $\sigma \models A \overset{+}{\rhd} G$. Also, $j \neq i$ and both x_k and x_r change in the step $\langle \sigma[0], \sigma[1] \rangle$. So, the behavior σ is noninterleaving. In other words, a realizable open system property includes behaviors with arbitrary interleaving, whatever the action μ. We have shown that there exists a behavior σ that is noninterleaving and satisfies $A \overset{+}{\rhd} G$.

Therefore, it would be inaccurate to call "interleaving" a property of the form $A \overset{+}{\rhd} G$. For this reason, when working with open systems, we will call "interleaving" the property $A \wedge G$, which describes the closed system that results when the environment satisfies the assumption.

In contrast, not all open system properties have shared state. There are choices of open system properties and μ that are disjoint-state.

5.2.4 Stepwise type precondition needed inside G

An open system property $A \overset{+}{\rhd} G$ can be unrealizable for some formulae G that we would expect it to be realizable. The cause is that even if G implies an initial condition assumed in A, it still fails to be realizable. We explain it below using an example.

Consider the definition of a game with two variables x and y. Let player 0 be the environment, and player 1 the component that we specify. The environment controls the variable x, and the component the variable y.

Proposition 7 (Unrealizable open system property)
ASSUME: *1.* CONSTANT f

2. VARIABLES x, y

3. $A \triangleq \wedge\ y = \text{FALSE}$
$\qquad\quad \wedge\ \Box(x = \text{TRUE})$

4. $G \triangleq \wedge\ \Box[\wedge\ y \in \text{BOOLEAN}\ ]_{\langle x,y \rangle}$
$\qquad\qquad\quad \wedge\ y' = x$
$\qquad\quad \wedge\ \Box\Diamond(y = x)$

5. $\varphi \triangleq A \overset{+}{\rhd} G$

6. $\psi \triangleq \Box[y' = f[\langle x, y \rangle]]_{\langle x,y \rangle}$

PROVE: $\exists_{\text{behavior}}\, \sigma : \psi \wedge \neg\varphi$

PROOF: on page 54.

By Proposition 7, the property φ is unrealizable. The unrealizability is due to how the operator $\overset{+}{\rhd}$ is defined. A similar proof can be given for a realization that includes a conjunct of the form $\mathrm{WF}_y(y' = f[\langle x, y \rangle])$ as defined in Fig. 6. In that case, proving that a realization exists from any initial condition relies on a machine closure argument [27, Prop.1, p.519], thus replacing Proposition 2.

The behavior σ does not satisfy $A \overset{+}{\rhd} G$, because the environment violated the initial condition ($\sigma[0][\![y \notin \text{BOOLEAN}\ ]\!]$). But $\mathit{PrefixSat}(\sigma, 0, A)$ ignores whether the state $\sigma[0]$ violates A, and requires only that the environment have *some* execution, from *some* initial state. However, $\mathit{PrefixSat}(\sigma, 1, G)$ requires that the component have an execution from the given state $\sigma[0]$. This is impossible, because the component is asked to continue from a state where the environment has violated its type invariant.

Another example The example of Proposition 7 involves initial states. A similar situation can arise during steps of a behavior. The violation of $A \overset{+}{\rhd} G$

by a step is related to the "last environment step" and the "next component step". Consider previous example modified with the following guarantee

$$G \triangleq \begin{array}{l} \land\ \Box[(y \in \text{BOOLEAN}) \land (y' = x)]_{\langle x,y \rangle} \\ \land\ \Box\Diamond(y = x) \\ \land\ \Box\Diamond(y = \text{TRUE}) \end{array}$$

Pick a behavior σ with

$$\begin{array}{l} \land\ \sigma[0][\![(x = \text{TRUE}) \land (y = \text{FALSE})]\!] \\ \land\ \sigma[1][\![(x \notin \text{BOOLEAN}) \land (y = \text{TRUE})]\!] \end{array}$$

The initial state satisfies the assumption A, so the predicate $\mathit{PrefixSat}(\sigma, 1, A)$ is true. The nonstuttering step $\langle \sigma[0], \sigma[1] \rangle$ satisfies the action of the safety conjunct in G. However, the step $\langle \sigma[0], \sigma[1] \rangle$ violates the safety assumption of A. This is ignored by the predicate $\mathit{PrefixSat}(\sigma, 1, A)$. The predicate $\mathit{PrefixSat}(\sigma, 2, G)$ is false. The reason is that indefinite stuttering from $\sigma[1]$ violates $\Box\Diamond(y = x)$, and any nonstuttering G-step leads to $y \notin \text{BOOLEAN}$, so indefinite stuttering that violates $\Box\Diamond(y = \text{TRUE})$. Thus, $A \overset{+}{\rhd}$ is violated.

The above examples arise because *PrefixSat* (Eq. (15) on page 34) requires that the component be able to continue from the *actual* state $\sigma[1]$. The change of variable y from state $\sigma[0]$ to $\sigma[1]$ is constrained, but the change of variable x is not, thus wild environment behavior prevents the component from satisfying the guarantee in $\mathit{PrefixSat}(\sigma, 2, G)$.

The definition of $\overset{+}{\rhd}$ could have included overwriting within *PrefixSat* of the values that uncontrolled variables take in state $\sigma[1]$. This would check what values the component assigned in state $\sigma[1]$ to variables that it controls. It would ignore the (yet) unchecked values of environment variables in state $\sigma[1]$, and require that there be a way for the component to continue, if it was allowed to overwrite those uncontrolled values in state $\sigma[1]$.

With the current definition of the operator $\overset{+}{\rhd}$ in TLA$^+$, we should include a "stepwise type precondition" in the next-state action of G. In our example, this means modifying G to

$$G \triangleq \begin{array}{l} \land\ \Box[\,(y \in \text{BOOLEAN}) \Rightarrow \begin{array}{l} \land\ y \in \text{BOOLEAN} \\ \land\ y' = x \end{array}]_{\langle x,y \rangle} \\ \land\ \Box\Diamond(y = x) \end{array}$$

Another way to avoid this situation with $\overset{+}{\rhd}$ is to define a slight variant of this operator, a modification that will be discussed elsewhere.

5.3 Synchronous games with full information

A "synchronous" (and "Moore") system with full information can be encoded using $\mu \triangleq \neg\text{UNCHANGED}\ \langle x, y, m \rangle$ in Fig. 6

$$\Box[\mu \Rightarrow A]_{\langle m,x,y \rangle} \equiv \Box[(\neg\text{UNCHANGED}\ \langle x, y, m \rangle) \Rightarrow A]_{\langle m,x,y \rangle} \equiv \Box[A]_{\langle m,x,y \rangle}$$

The y in the subscript $\langle m, x, y\rangle$ ensures that the strategy f does not miss any steps that change the environment variable y. This represents a "synchronous" arrangement between the component and its environment. A "Mealy" component can read the value y', suggesting modifying Fig. 6 to

$$\begin{array}{rcl} A & \triangleq & \land\ x' = f[\langle m, x, y, y'\rangle] \\ & & \land\ m' = g[\langle m, x, y, y'\rangle] \end{array}$$

An "asynchronous" arrangement is represented by omitting variable y from the subscript, by letting $\neg\mu$ allow steps that the environment changes, but the system stutters $\mu \triangleq \neg\text{UNCHANGED } \langle m, x\rangle$

$$\Box[\mu \Rightarrow A]_{\langle m,x,y\rangle} \equiv \Box[(\neg\text{UNCHANGED } \langle m, x\rangle) \Rightarrow A]_{\langle m,x,y\rangle} \equiv \Box[A]_{\langle m,x\rangle}$$

The environment can take as many nonstuttering steps as it likes between consecutive nonstuttering system steps. The system does not observe y during those environment steps. This mathematical model corresponds to those games of partial information that Pnueli calls "asynchronous" [29]. There is little point in writing a "Mealy" form, because the component already does not participate in some steps that the environment changes. Expressing the above cases using the same module demonstrates that the proposed definition is reusable for defining a range of different flavors of game.

Another case we can define is a synchronous game of partial information. This case can be represented by hiding an environment variable y in φ, or by omitting the variable y from the arguments of f. In either approach, the system knows that y is a declared variable that changes at certain steps, but cannot observe the value of y. The environment takes as many nonstuttering steps as the system,, but the system cannot observe some changes of the environment.

6 Discussion

6.1 Specifying a type invariant is not restrictive

In Section 3.2, we showed that the formula φ should be satisfied if x, h do not eventually enter a set. Based on this, we concluded that φ should be satisfied whenever x, h start outside some specific set.

A further question is whether φ should also contain type invariants. If the formula φ does not contain enough type invariants, then even the behaviors of only variables that occur in φ can form a proper class (using the projection operator defined on 65). In other words, even the sequences that matter are so many that they don't fit in a set. We expect that this makes automated reasoning more difficult.

In contrast, if the formula φ contains enough type invariants, when the environment satisfies the assumption, the projected behaviors that could be winning form a set. It should then suffice to reason about a set of sequences at the semantic level.

In the presence of type invariants that restrict variable values to sets, proving that a program implements realizability for some fragment of formulae φ (for example, GR(1) [3]) will involve reasoning about the type invariants. So, the constructive reasoning has been shifted to writing proofs, rather than rendering the definition of realizability less readable.

Symbolic synthesis algorithms need to select finite representations for variables. Type invariants in φ can be used as syntactic "type hints" for these algorithms, in the spirit of type annotations for type checking [5] and type synthesis [14].

6.2 Variants for defining realizability

No set of all strategies In general, any function could be a candidate strategy to select as f or g. The collection of all functions is a proper class [13, p.23], not a set. So, the collection of all candidate solutions is not a set.

Axiomatic or constructive? If φ contains suitable type invariants, then a less axiomatic (more constructive) definition of synthesis can be given too. In this case, it suffices for any realizing strategy f, g to take arguments and values from some set S (S is defined using sets mentioned in the type invariants of φ). It then suffices to search for implementations $f, g \in Q \triangleq [S \times S \times Nat \rightarrow S]$. But, to write Q, we must define S, thus mention in the implementation ψ the sets that appear in the type invariants of φ. This approach leads to a definition of realizability that depends on how the formula φ is written.

Functions as environment strategies? We did not represent the environment behavior using functions, only the system's behavior. If we did, then we would need to quantify universally over environment functions (possibly from some set of functions, in presence of type invariants). This would yield a less axiomatic definition, but with some additional differences.

A behavior may initially satisfy the assumption's type invariant, but later assign arbitrary values to environment variables, and so violate the assumption. A given environment strategy function cannot assign arbitrary values (its range is a set), so it cannot produce such a behavior. So, this behavior will be missing from what f, g must be able to cope with.

A workaround is to let the environment strategy function take as value an auxiliary value that violates the assumption's type invariant. Such a function can produce behaviors similar to those described above, but with less "variety" of values violating the typeness assumption.

A (countable) infinite number of variable names are definable in TLA$^+$. Realizability must be defined using a module with finitely many characters. So, an environment strategy would only assign to some environment variables. The rest would still behave arbitrarily. Finitely many variables occur in the formula φ, so an environment strategy would still assign to sufficiently many variables.

A function of history? Lamport's definition (Appendix C) uses a history variable that records all variables that occur in the specification. We used a function on an uncontrolled variable h without requiring that h record history. It is still possible to make h a history variable, by incorporating suitable assumptions in φ. A motivation for not letting h record infinite history is that we want to restrict functions to finite memory, and also consider synthesis of g, the memory update function. An f that reads as input an infinite history record h would have to be a function with infinite domain, so conceptually a little different from its real-world implementation.

6.3 Interpretations and models

Pitfalls of modifying an unstructured interpretation Modifying an interpretation I_A to obtain another interpretation I_B is difficult in absence of a recursive definition. Let the identifier "m" be declared as a constant. Suppose that we pick some fixed value u, and define

$$I_B(e, s) \triangleq \text{IF } e = \text{“}m\text{”} \text{ THEN } u \text{ ELSE } I_A(e, s)$$

If our purpose was to ensure that $I_B(\text{“m”}, s) = u$, then it seems to have been accomplished. However, in carelessly modifying I_A, we could have violated a TLA$^+$ axiom. How can this happen? Suppose that the u we picked results in $I_B(\text{“}\mathit{IsAFunction}(m)\text{”}, s)$. So, we created an additional function symbol in the fresh model I_B. Also, suppose that there was an identifier "p" such that

$$\begin{aligned}
&\wedge\ I_A(\text{“}p\text{”}, s) = u \\
&\wedge\ I_A(\text{“}\mathit{IsAFunction}(p)\text{”}, s) \\
&\wedge\ I_A(\text{“DOMAIN } p\text{”}, s) \neq I_A(\text{“DOMAIN } m\text{”}, s)
\end{aligned}$$

We did not change the interpretation of these expressions in I_B, so

$$\begin{aligned}
&\wedge\ I_B(\text{“}p = m\text{”}, s) \\
&\wedge\ I_B(\text{“}\mathit{IsAFunction}(p)\text{”}, s) \\
&\wedge\ I_B(\text{“DOMAIN } p\text{”}, s) \neq I_B(\text{“DOMAIN } m\text{”}, s)
\end{aligned}$$

This violates the axiom about function equality [8, p.303] (see also axiom schema in Section 3.2.3)

$$\begin{aligned}
\models_{I_B}\quad & \mathit{IsAFunction}(p) \wedge \mathit{IsAFunction}(m) \Rightarrow \\
& (p = m) \equiv \wedge\ \text{DOMAIN } p = \text{DOMAIN } m \\
& \qquad\qquad\quad \wedge\ \forall x \in \text{DOMAIN } p : p[x] = m[x].
\end{aligned}$$

So, the resulting interpretation I_B is *not* a model of TLA$^+$.

Too specific a model In our first attempt to define an interpretation for function application syntax, we used Hilbert's ε operator similarly to [6, p.4] so that $f[x]$ be interpreted as follows

$$\mathit{Apply}(f, x) \triangleq \text{CHOOSE } y : \{x, \{y, x\}\} \in f$$

This definition would lead to an interpretation of TLA$^+$ too restrictive for the modifications that we make to derive a new interpretation. The undesired restriction is that there is one common value that all functions take outside their domain

$$\begin{aligned}
&\forall f, x : x \notin \text{DOMAIN } f \Rightarrow \\
&\qquad \neg \exists\, y : \{x, \{x, y\}\} \in f \Rightarrow \\
&\qquad (\text{CHOOSE } y : \{x, \{y, x\}\} \in f = \text{CHOOSE } y : \text{FALSE}) \Rightarrow \\
&\qquad \mathit{Apply}(f, x) = \text{CHOOSE } y : \text{FALSE}
\end{aligned}$$

which follows from the axiomatic definition of CHOOSE [8, (16.2), p.295].

This problem can be avoided by using the operator *Choice* defined in [8, p.295]. *Choice* is again defined in terms of CHOOSE, so the freedom to select what $f[x]$ means will again reduce to how constant operators are defined. So, we will not use *Choice*, but define what $f[x]$ means with the semantic interpretation of TLA$^+$.

Deeper nesting We defined a relative interpretation of the TLA$^+$ syntax for function application, $f[x]$, in the metatheory ZF. This interpretation was defined using a recursive definition of the *Flatten* operator in ZF. An operator is a formula (a syntactic entity), so arguing that it makes sense takes place in the metatheory (of ZF) using finitistic arguments [13].

We could have interpreted TLA$^+$ expressions of the form $f[x]$ as ZF expressions of the same form (assuming function syntax within ZF). With this approach, to modify what $f[x]$ means in a reinterpretation of TLA$^+$ in ZF, we would need to interpret ZF in ZF, and modify the interpretation of $f[x]$. This involves arguing in the metatheory of the metatheory of TLA$^+$, and keeping track of the distinction between three levels. We avoided this nested approach to make the proof easier to read.

7 Conclusion

We formalized a definition of synthesis based on [1]. In the process, we made some specification choices and discussed alternatives. We showed that initial conditions should appear in the form of set containment assumptions to avoid unprovability of realizability.

Proving a synthesis algorithm correct will require reasoning about type invariants from the property to be implemented. We kept the definition simpler, by not mentioning type invariants.

The definition of realizability proposed here is suitable for representing different types of games, including what have been known as asynchronous games, and games with partial information.

Acknowledgment The authors would like to thank Scott Livingston for providing feedback on drafts of this manuscript, and Necmiye Özay for interesting

discussions related to synthesis.

This work was supported in part by the TerraSwarm Research Center, one of six centers supported by the STARnet phase of the Focus Center Research Program (FCRP), a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

A Proofs

Caution: Several proof steps are carried out within the metatheory, affecting whether deduction steps are to be understood within TLA$^+$ (where the deduction principle does not hold [26, Sec.8.2, p.37]), or within ZF (where the deduction principle holds). We use $\textsc{zf new}\ u$ to emphasize that u is a constant in the metatheory. Behaviors, states, $\sigma \models F$ live in the metatheory.

Proof rule names are abbreviated as: ∀E: universal elimination, ∀G: universal generalization, ∃E: existential elimination, ∃G: existential generalization [18, Thm.II.16, p.48], ∧E: conjunction elimination, ∧I: conjunction introduction, ∨I: disjunction introduction, MP: modus ponens [18, p.40], DP: deduction principle [18, Thm.II.5, p.42]. The proof rules STL1–STL6, TLA1, TLA2, INV1, INV2 are defined in [30, Fig.5, p.888].

Let *VarNames* be the collection of all variable names, which is a set [8, pp.311, 313]. The following two operators characterize states and behaviors

$$\begin{aligned} \mathit{IsAState}(s) &\triangleq \land\ \mathit{IsAFunction}(s) \\ &\ \land\ \textsc{domain}\ s = \mathit{VarNames} \\ \mathit{IsABehavior}(\sigma) &\triangleq \land\ \mathit{IsAFunction}(\sigma) \\ &\ \land\ \textsc{domain}\ \sigma = \mathit{Nat} \\ &\ \land\ \forall\, n \in \mathit{Nat} : \mathit{IsAState}(\sigma[n]) \end{aligned} \tag{9}$$

We define $=_x$ for states

$$\begin{aligned} s =_x t &\triangleq s[\text{“}x\text{”}] = t[\text{“}x\text{”}] \\ s =_{x,\dots,z} t &\triangleq \land\ s =_x t \\ &\ \ \vdots \\ &\ \land\ s =_z t \\ s \neq_{x,\dots,z} t &\triangleq \neg s =_{x,\dots,z} t \end{aligned} \tag{10}$$

These definitions are similar to those in [31]. Note that x is the only variable included in the comparison, which is opposite to the definition of $=_x$ in [30, p.903]. Let

$$\mathit{EqualUpTo}(\mathit{var}, \tau, \rho) \triangleq \forall\, n \in \mathit{Nat} : \forall\, v \in \mathit{VarNames} \setminus \{\mathit{var}\} : \tau[n][v] = \rho[n][v] \tag{11}$$

The operator *EqualUpTo* corresponds to $=_x$ as defined (for behaviors) in [30].

The "unstuttering" operator $\natural$ is defined as [8, p.311]

$$\begin{array}{lll} \natural\sigma & \triangleq & \text{LET } f[n \in \mathit{Nat}] \triangleq \text{ IF } n = 0 \text{ THEN } 0 \\ & & \qquad\qquad\qquad\qquad\qquad \text{ELSE IF } \sigma[n] = \sigma[n-1] \\ & & \qquad\qquad\qquad\qquad\qquad\qquad \text{THEN } f[n-1] \\ & & \qquad\qquad\qquad\qquad\qquad\qquad \text{ELSE } f[n-1] + 1 \\ & & \qquad S \triangleq \{f[n] : n \in \mathit{Nat}\} \\ & & \text{IN} \quad [n \in S \mapsto \sigma[\text{CHOOSE } i \in \mathit{Nat} : f[i] = n] \end{array}$$

If a behavior σ has an (infinite) stuttering "tail", then $\mathit{IsFiniteSet}(\text{DOMAIN } \natural\sigma)$ with the definition of [8, p.311] (above). In this case, the definition of [30, Eq.(48), p.904] preserves the stuttering tail, so $\neg\mathit{IsFiniteSet}(\text{DOMAIN } \natural\sigma)$. For the purpose of defining $\boldsymbol{\exists}$, it does not make a difference which definition of $\natural$ is used.

The operator $\boldsymbol{\exists}$ can be defined as [30, Eq.(49)]

$$\sigma \models \boldsymbol{\exists}\, x : F \;\triangleq\; \exists \tau, \rho : \begin{array}{l} \land\; \mathit{IsABehavior}(\tau) \\ \land\; \mathit{IsABehavior}(\rho) \\ \land\; \natural\rho = \natural\sigma \\ \land\; \mathit{EqualUpTo}(\text{"}x\text{"}, \rho, \tau) \\ \land\; \tau \models F \end{array} \tag{12}$$

We define the operator $\sim_x$ as follows ($\sim_x$ is not defined in [30])

$$\sigma \sim_x \tau \;\triangleq\; \exists \rho : \begin{array}{l} \land\; \mathit{IsABehavior}(\rho) \\ \land\; \natural\rho = \natural\sigma \\ \land\; \mathit{EqualUpTo}(\text{"}x\text{"}, \rho, \tau) \end{array} \tag{13}$$

Substituting Eq. (13) in Eq. (12) we obtain

$$\begin{aligned} \sigma \models \boldsymbol{\exists}\, x : F &= \exists \tau : \begin{array}{l} \land\; \mathit{IsABehavior}(\tau) \\ \land\; \sigma \sim_x \tau \\ \land\; \tau \models F \end{array} \\ &= \exists_{\text{behavior}}\, \tau : \begin{array}{l} \land\; \sigma \sim_x \tau \\ \land\; \tau \models F \end{array} \end{aligned} \tag{14}$$

If $\mathit{IsABehavior}(\sigma)$, then Eq. (14) can be shown to be equivalent to the definition of $\boldsymbol{\exists}$ in [8, p.316] with the operator $\sim_x$ defined as in [25, p.4].

As defined in Eq. (1),

$$\begin{aligned} \forall_{\text{behavior}}\, \sigma : F &\triangleq \neg\, \exists_{\text{behavior}}\, \sigma : \neg F \\ &= \neg\, \exists\, \sigma : \mathit{IsABehavior}(\sigma) \land \neg F \\ &= \forall\, \sigma : \neg\big(\mathit{IsABehavior}(\sigma) \land \neg F\big) \\ &= \forall\, \sigma : \mathit{IsABehavior}(\sigma) \Rightarrow F \end{aligned}$$

Remark 8 (Tuple constructor syntax) *We use a schematic definition of tuples, instantiated as*

$$\langle x, h, m \rangle \triangleq [i \in \{1,2,3\} \mapsto \text{IF } i = 1 \text{ THEN } x \ \text{ELSE IF } i = 2 \text{ THEN } h \ \text{ELSE } m]$$

The reason is that the definition given in [8, §16.1.9] refers to the operator $\leq$, which is defined in the module ProtoReals [8, Fig.18.5, pp.346–347], which in turn contains tuple syntax (for example, $\langle a, b \rangle \in Leq$).

The conditional expression above has been used to define tuples also in [31, p.434], where the domain is defined using the operator $\leq$.

Equivalently, we could have used throughout the entire proof of Lemma 6 function constructor syntax, in place of the expression $\langle x, h, m \rangle$.

An open system deals with an environment it cannot control [9, 2, 3]. It can succeed only if the environment behaves as assumed, which can be described with a property of the form $A \stackrel{+}{\rhd} G$ [8, p.156, p.316], [25, p.4]. This property requires the system to exhibit behavior that satisfies G (guarantee), as long as the environment does not violate A (assumption). We can write

$$A = Init_A \land A \qquad G = Init_G \land G$$

to emphasize the initial predicates $Init_A, Init_G$ of the assumption and guarantee. It follows that an open-system specification has the form

$$(Init_A \land A) \stackrel{+}{\rhd} (Init_G \land G)$$

For convenience, let *PrefixSat* [8, p.316] be defined with σ as an explicit argument

$$(15) \qquad PrefixSat(\sigma, n, H) \triangleq \exists_{\text{behavior}}\, \tau : \begin{array}{l} \land\ \forall\, i \in 0..(n-1) : \tau[i] = \sigma[i] \\ \land\ \tau \models H \end{array}$$

The operator $\stackrel{+}{\rhd}$ is defined as [8, p.316, equiv. p.337]
(16)

$$\sigma \models A \stackrel{+}{\rhd} G \triangleq \begin{array}{l} \land\ \sigma \models A \Rightarrow G \\ \land\ \forall\, n \in Nat : PrefixSat(\sigma, n, A) \Rightarrow PrefixSat(\sigma, n+1, G) \end{array}$$

[Proposition 3 on page 5] PROOF:
$\langle 1 \rangle 1.$ SUFFICES: ASSUME: $IsABehavior(\sigma)$
PROVE: $\exists_{\text{behavior}}\, \tau : \begin{array}{l} \land\ \sigma \sim_x \tau \\ \land\ \tau \models \begin{array}{l} \land\ x = u \\ \land\ \Box[x' = f[\langle x, h \rangle]]_{\langle x, h \rangle} \end{array} \end{array}$
BY DEF $\forall_{\text{behavior}}$.
$\langle 1 \rangle 2.$ DEFINE *VariableNames* as the set of all variable names
BY [8, p.311], this set exists.

$\langle 1\rangle 3.$ DEFINE

$$\begin{aligned} \mathit{TauN}(\tau, n) &\triangleq \text{ IF } n = 0 \\ &\quad \text{ THEN } [\sigma[0] \text{ EXCEPT } ![\![x]\!] = \sigma[0][\![u]\!]] \\ &\quad \text{ ELSE } [\sigma[n] \text{ EXCEPT } ![\![x]\!] = \tau[n-1][\![f[\langle x, h\rangle]]\!]]] \\ \mathit{TauFromSigma}(\tau) &\triangleq \tau = [n \in \mathit{Nat} \mapsto \mathit{TauN}(\tau, n)] \end{aligned}$$

$\langle 1\rangle 4.$ $\exists\, \tau : \mathit{TauFromSigma}(\tau)$
  $\langle 2\rangle 1.$ SUFFICES: ASSUME: $\forall_{\text{behavior}}\, \tau : \neg\, \mathit{TauFromSigma}(\tau)$
      PROVE: FALSE
  $\langle 2\rangle 2.$ PICK $\tau : \mathit{IsABehavior}(\tau) \land \neg\, \mathit{TauFromSigma}(\tau)$
  $\langle 2\rangle 3.$

$$\begin{aligned} &\forall_{\text{behavior}}\, \tau : \neg\, \mathit{TauFromSigma}(\tau) \equiv \\ &\forall_{\text{behavior}}\, \tau : \exists\, n \in \mathit{Nat} : \tau[n] \neq \mathit{TauN}(\tau, n) \end{aligned}$$

  $\langle 2\rangle 4.$ $\exists\, n \in \mathit{Nat} : \tau[n] \neq \mathit{TauN}(\tau, n)$
  $\langle 2\rangle 5.$ $\forall\, n \in \mathit{Nat} : \mathit{IsAState}(\mathit{TauN}(\tau, n))$
    BY $\langle 1\rangle 3$, Induction, $\langle 2\rangle 2$.
  $\langle 2\rangle 6.$ $\exists_{\text{behavior}}\, g : g = [n \in \mathit{Nat} \mapsto \mathit{TauN}(\tau, n)]$
    BY $\langle 2\rangle 5$
  $\langle 2\rangle 7.$ $\exists_{\text{behavior}}\, g : \forall\, n \in \mathit{Nat} : g[n] = \mathit{TauN}(g, n)$
    BY $\langle 2\rangle 6$, Induction.
  $\langle 2\rangle 8.$ Q.E.D.
    BY $\langle 2\rangle 7$ and ASSUMPTION of $\langle 2\rangle 1$.
$\langle 1\rangle 5.$ PICK $\tau : \mathit{TauFromSigma}(\tau)$
  BY $\langle 1\rangle 4$.
$\langle 1\rangle 6.$ $\mathit{IsABehavior}(\tau)$
  $\langle 2\rangle 1.$ $\mathit{IsAFunction}(\tau)$
    BY $\langle 1\rangle 3$, $\langle 1\rangle 5$.
  $\langle 2\rangle 2.$ DOMAIN $\tau = \mathit{Nat}$
    BY $\langle 1\rangle 3$, $\langle 1\rangle 5$.
  $\langle 2\rangle 3.$ $\forall\, n \in \mathit{Nat} : \mathit{IsAState}(\tau[n])$
    BY DEF *IsAState* in [8, p.313], $\langle 1\rangle 3$, $\langle 1\rangle 5$.
  $\langle 2\rangle 4.$ Q.E.D.
    BY $\langle 2\rangle 1$, $\langle 2\rangle 2$, $\langle 2\rangle 3$.
$\langle 1\rangle 7.$ $\sigma \sim_x \tau$
  $\langle 2\rangle 1.$ $\mathit{EqualUpTo}(\text{“}x\text{”}, \sigma, \tau)$
    BY $\langle 1\rangle 3$, $\langle 1\rangle 5$, $\langle 1\rangle 1$, $\langle 1\rangle 6$, DEF *EqualUpTo* (Eq. (11) on page 32).
  $\langle 2\rangle 2.$ $\land\ \mathit{IsABehavior}(\sigma)$
      $\land\ \natural\sigma = \natural\sigma$
    BY $\langle 1\rangle 1$, AXIOM equality reflexive.
  $\langle 2\rangle 3.$ Q.E.D.
    BY $\langle 2\rangle 1$, $\langle 2\rangle 2$, DEF $\sim_x$ (Eq. (13) on page 33), $\exists$G with WITNESS $\rho \triangleq \sigma$.
$\langle 1\rangle 8.$ $\tau \models (x = u)$
  BY $\langle 1\rangle 3$, $\langle 1\rangle 5$.

$\langle 1 \rangle 9.$ $\tau \models \Box[x' = f[\langle x, h \rangle]]_{\langle x,h \rangle}$
BY $\langle 1 \rangle 3$, $\langle 1 \rangle 5$, induction, $f[\langle x, h \rangle]$ is a syntactically well-formed expression, TLA$^+$ is untyped, DEF of $\Box[A]_v$, ASSUMPTION 1, 2 the expression f is a constant symbol, h, x variable symbols, so symbol x does not occur in the expressions f, h.
$\langle 1 \rangle 10.$ Q.E.D.
BY $\langle 1 \rangle 6$, $\langle 1 \rangle 7$, $\langle 1 \rangle 8$, $\langle 1 \rangle 9$.

[Lemma 4 on page 5] PROOF:
$\langle 1 \rangle 1.$ $\exists\, u, r : (\langle x, h \rangle = \langle u, r \rangle) \Rightarrow \neg\varphi(x, h)$
$\langle 2 \rangle 1.$ $\exists\, u, r : \mathit{Init}(u, r) \equiv$ FALSE
BY ASSUMPTION 6.
$\langle 2 \rangle 2.$ $\neg \mathit{Init}(x, h) \Rightarrow \neg\varphi(x, h)$
BY contrapositive of ASSUMPTION 5.
$\langle 2 \rangle 3.$ Q.E.D.
BY $\langle 2 \rangle 1$, $\langle 2 \rangle 2$
$\langle 1 \rangle 2.$ $\forall\, u : \boldsymbol{\forall}\, h : \boldsymbol{\exists}\, x : (x = u) \wedge \psi(x, h, f)$
BY Proposition 2 and a formula being valid defined as its universal closure being TRUE.
$\langle 1 \rangle 3.$ $\forall\, u, r : \boldsymbol{\exists}\, x, h : (\langle x, h \rangle = \langle u, r \rangle) \wedge \psi(x, h, f)$
$\langle 1 \rangle 4.$ PICK $u, r : (\langle x, h \rangle = \langle u, r \rangle) \Rightarrow \neg\varphi(x, h)$
BY $\langle 1 \rangle 1$
$\langle 1 \rangle 5.$ $\boldsymbol{\exists}\, x, h : (\langle x, h \rangle = \langle u, r \rangle) \wedge \psi(x, h, f)$
BY $\langle 1 \rangle 3$
$\langle 1 \rangle 6.$ Q.E.D.
BY $\langle 1 \rangle 5$, $\langle 1 \rangle 4$ and take universal closure with respect to f.

[Proposition 5 on page 11] PROOF SKETCH:
$\langle 1 \rangle 1.$ The truthness of axioms and proof rules of TLA$^+$ is independent of what value $f[x]$ is for $x \notin$ DOMAIN f.
PROOF SKETCH: BY the moderate interpretation of Boolean operators [8, p.297], if $f[x]$ appears in an axiom, then the truthness of the axiom is independent of what value $f[x]$ is for $x \notin$ DOMAIN f [8, §16.1.7]. In other words, if f is a function symbol, then the axioms of TLA$^+$ leave the values $f[x]$ for $x \notin$ DOMAIN f completely unspecified.
BY ASSUMPTION 2 the function constructor syntax $[x \in S \mapsto e(x)]$ satisfies the relevant axioms, which can be proved using the dependence of *MakeFunction* only on values of $\mathit{ApplyFunc}(p, \_)$ inside $\mathit{DomainOf}(p)$. This ensures that the function constructor axiom of [25] remains true.
$\langle 1 \rangle 2.$ The modification of the interpretation from J to R changes only values $f[x]$ for $x \notin$ DOMAIN f.
PROOF SKETCH: BY ASSUMPTION 3, 4.
$\langle 1 \rangle 3.$ The modification of the interpretation from J to R cannot affect the truthness of axioms or proof rules.
$\langle 1 \rangle 4.$ The axioms and proof rules of TLA$^+$ are true in the interpretation J, σ.
$\langle 2 \rangle 1.$ The TLA$^+$ theorems and proof rules are true in the interpretation J, σ.
BY ASSUMPTION 1.

⟨2⟩2. Any axiom is a theorem.
⟨2⟩3. Q.E.D.
BY ⟨2⟩1, ⟨2⟩2.
⟨1⟩5. The axioms and proof rules of TLA$^+$ are true in the interpretation R, σ.
BY ⟨1⟩4, ⟨1⟩3, MP.
⟨1⟩6. All theorems of TLA$^+$ are true in R, σ.
BY ⟨1⟩5, induction from axioms applying proof rules.
⟨1⟩7. Q.E.D.
BY ⟨1⟩6, DEF *IsAModel*.

[Lemma 6 on page 12] PROOF:
⟨1⟩1. DEFINE:

$$F \triangleq \forall f : \mathit{IsAFunction}(f) \Rightarrow \boldsymbol{\exists}\, x, h : \psi(x, h, f) \land \neg\varphi(x, h)$$

(Note that $F = \forall f : \boldsymbol{\exists}\, x, h : \mathit{IsAFunction}(f) \Rightarrow \psi(x, h, f) \land \neg\varphi(x, h))$
⟨1⟩2. $\models \forall f : \mathit{IsAFunction}(f) \Rightarrow \mathit{CanLoseOutside}$
⟨2⟩1. $\models \forall f : \mathit{IsAFunction}(f) \Rightarrow \text{TRUE}$
BY predicate logic.
⟨2⟩2. $\models \mathit{CanLoseOutside}$
BY ASSUMPTION 10, ⟨2⟩1.
⟨2⟩3. Q.E.D.
BY ⟨2⟩1, ⟨2⟩2.
⟨1⟩3. DEFINE: 1. $\mathit{FirstTwo}(S) \triangleq \{\langle a, b\rangle : \exists\, c : \langle a, b, c\rangle \in S\}$
2. $\mathit{BehOutside} \triangleq \forall f : \lor \neg \mathit{IsAFunction}(f)$
$\qquad \lor \boldsymbol{\exists}\, x, h : \land \neg\varphi(x, h)$
$\qquad\qquad \land \Box(\langle x, h\rangle \notin \mathit{FirstTwo}(\text{DOMAIN } f))$
(Note that $\mathit{BehOutside} = \forall f : \boldsymbol{\exists}\, x, h : \mathit{IsAFunction}(f) \Rightarrow$
$\land \neg\varphi(x, h)$)
$\land \Box(\langle x, h\rangle \notin \mathit{FirstTwo}(\text{DOMAIN } f))$)
⟨1⟩4. $\models \mathit{BehOutside}$
⟨2⟩1. $\models \forall f : \lor \neg\mathit{IsAFunction}(f)$
$\qquad \lor \forall S : \boldsymbol{\exists}\, x, h : \land \neg\varphi(x, h)$
$\qquad\qquad \land \Box(\langle x, h\rangle \notin S)$
BY ASSUMPTION 9, ⟨1⟩2.
⟨2⟩2. $\models \forall S, f : \lor \neg\mathit{IsAFunction}(f)$
$\qquad \lor \boldsymbol{\exists}\, x, h : \land \neg\varphi(x, h)$
$\qquad\qquad \land \Box(\langle x, h\rangle \notin S)$
BY ⟨2⟩1, symbol S does not occur in the expression $\mathit{IsAFunction}(f)$.
⟨2⟩3. $\models \forall f : \lor \neg\mathit{IsAFunction}(f)$
$\qquad \lor \boldsymbol{\exists}\, x, h : \land \neg\varphi(x, h)$
$\qquad\qquad \land \Box(\langle x, h\rangle \notin \mathit{FirstTwo}(\text{DOMAIN } f))$
BY ⟨2⟩2, ∀E with $S \triangleq \mathit{FirstTwo}(\text{DOMAIN } f)$.
⟨2⟩4. Q.E.D.
BY ⟨2⟩3, ⟨1⟩4/2.

$\langle 1\rangle 5.$ $\models \boldsymbol{\forall}\, x, h : \forall f : \boldsymbol{\exists}\, m : \begin{array}{l} \wedge\ m = \langle f, 0\rangle \\ \wedge\ \Box[m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle} \end{array}$

  $\langle 2\rangle 1.$ $\boldsymbol{\exists}\, m : \begin{array}{l} \wedge\ m = \langle f, 0\rangle \\ \wedge\ \Box[\begin{array}{l}\wedge\ m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle} \\ \wedge\ \langle x, h\rangle' \neq \langle x, h\rangle \end{array} \end{array}$

  BY [10, §2.4] (m is a history-determined variable).

  $\langle 2\rangle 2.$ $\models \Box[\begin{array}{l}\wedge\ m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle} \\ \wedge\ \langle x, h\rangle' \neq \langle x, h\rangle\end{array} \;\Rightarrow\; \Box[m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle}$

    $\langle 3\rangle 1.$ $\models [\begin{array}{l}\wedge\ m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle} \\ \wedge\ \langle x, h\rangle' \neq \langle x, h\rangle\end{array} \;\Rightarrow\; [m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle}$

    BY $\wedge$E.

    $\langle 3\rangle 2.$ Q.E.D.

    BY $\langle 3\rangle 1$, TLA2.

  $\langle 2\rangle 3.$ $\models \left(\begin{array}{l} \wedge\ m = \langle f, 0\rangle \\ \wedge\ \Box[\begin{array}{l}\wedge\ m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle} \\ \wedge\ \langle x, h\rangle' \neq \langle x, h\rangle\end{array} \end{array}\right) \Rightarrow \left(\begin{array}{l} \wedge\ m = \langle f, 0\rangle \\ \wedge\ \Box[m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle} \end{array}\right)$

  BY $\langle 2\rangle 2$.

  $\langle 2\rangle 4.$ Q.E.D.

  BY $\langle 2\rangle 1$, $\langle 2\rangle 3$, $(\models A \Rightarrow B) \Rightarrow (\models (\boldsymbol{\exists}\, m : A) \Rightarrow (\boldsymbol{\exists}\, m : B))$.

$\langle 1\rangle 6.$ 1. $W(m, x, h, f) \;\triangleq\; \begin{array}{l} \vee\ \neg \mathit{IsAFunction}(f) \\ \vee\ \begin{array}{l} \wedge\ \neg\varphi(x, h) \\ \wedge\ m = \langle f, 0\rangle \\ \wedge\ \Box[m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle} \\ \wedge\ \Box\big(\langle x, h\rangle \notin \mathit{FirstTwo}(\textsc{domain}\ f)\big) \end{array} \end{array}$

  2. $V(f) \;\triangleq\; \boldsymbol{\exists}\, x, h, m : W(m, x, h, f)$

  3. $Q \;\triangleq\; \forall f : V(f)$

$\langle 1\rangle 7.$ $\models Q$

  $\langle 2\rangle 1.$ $\models \forall f : \boldsymbol{\exists}\, x, h : \begin{array}{l} \vee\ \neg \mathit{IsAFunction}(f) \\ \vee\ \begin{array}{l} \wedge\ \neg\varphi(x, h) \\ \wedge\ \Box(\langle x, h\rangle \notin \mathit{FirstTwo}(\textsc{domain}\ f)) \end{array} \end{array}$

  BY $\langle 1\rangle 4$, $\langle 1\rangle 3/2$.

  $\langle 2\rangle 2.$ $\boldsymbol{\forall}\, x, h : \forall f : \boldsymbol{\exists}\, m : \begin{array}{l} \wedge\ m = \langle f, 0\rangle \\ \wedge\ \Box[m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle} \end{array}$

  BY $\langle 1\rangle 5$.

  $\langle 2\rangle 3.$ Q.E.D.

  BY $\langle 2\rangle 1$, $\langle 2\rangle 2$, ASSUMPTION 6.

$\langle 1\rangle 8.$ $J, \sigma \models Q$

  $\langle 2\rangle 1.$ $\mathit{IsAModel}(J, \sigma)$

  BY ASSUMPTION 4.

  $\langle 2\rangle 2.$ $\models Q$

  BY $\langle 1\rangle 7$.

  $\langle 2\rangle 3.$ Q.E.D.

  BY $\langle 2\rangle 1$, $\langle 2\rangle 2$, DEF $\models$ [18, p.22].

$\langle 1\rangle 9.\ \Gamma(\tau, u) \ \triangleq\ \land\ \mathit{IsABeh}(\tau)$
$\qquad\qquad\qquad\quad \land\ \sigma \sim_{x,h,m} \tau$
$\qquad\qquad\qquad\quad \land\ J|_{f \triangleq u}, \tau \models W(x, h, m, f)$

$\langle 1\rangle 10.\ \mathit{PickLosingBeh}(u) \ \triangleq\ \textsc{choose}\ \tau : \Gamma(\tau, u)$

$\langle 1\rangle 11.\ \forall\, u : \Gamma(\mathit{PickLosingBeh}(u), u)$

$\quad\langle 2\rangle 1.\ \forall\, u : \exists\, \tau : \Gamma(\tau, u)$

$\qquad\langle 3\rangle 1.\ J, \sigma \models \forall f : \boldsymbol{\exists}\, x, h, m : W(x, h, m, f)$

$\qquad\quad$ BY $\langle 1\rangle 6$, $\langle 1\rangle 7$.

$\qquad\langle 3\rangle 2.\ \forall\, u : J|_{f \triangleq u}, \sigma \models \boldsymbol{\exists}\, x, h, m : W(x, h, m, f)$

$\qquad\quad$ BY $\langle 3\rangle 1$, DEF $\forall$ (Eq. (4) on page 9).

$\qquad\langle 3\rangle 3.\ \forall\, u : \exists\, \tau : \land\ \mathit{IsABeh}(\tau)$
$\qquad\qquad\qquad\qquad\quad \land\ \sigma \sim_{x,h,m} \tau$
$\qquad\qquad\qquad\qquad\quad \land\ J|_{f \triangleq u}, \tau \models W(x, h, m, f)$

$\qquad\quad$ BY $\langle 3\rangle 2$, DEF $\boldsymbol{\exists}$.

$\qquad\langle 3\rangle 4.$ Q.E.D.

$\qquad\quad$ BY $\langle 3\rangle 3$, $\langle 1\rangle 9$.

$\quad\langle 2\rangle 2.\ \forall\, u : \Gamma(\textsc{choose}\ \tau : \Gamma(\tau, u), u)$

$\qquad$ BY $\langle 2\rangle 1$, CHOOSE AXIOM .

$\quad\langle 2\rangle 3.$ Q.E.D.

$\qquad$ BY $\langle 2\rangle 2$, $\langle 1\rangle 10$.

$\langle 1\rangle 12.\ \mathit{IsIndex}(i, \eta, u, v) \ \triangleq\ v = \eta[i][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq p}}$

$\langle 1\rangle 13.$

```
ApplyFunc_R(u, v)  ≜  LET
  $\eta \triangleq \mathit{PickLosingBeh}(u)$
  $k \triangleq \textsc{choose}\ i \in \mathit{Nat} : \land\ \mathit{IsIndex}(i, \eta, u, v)$
                                  $\land\ \neg\mathit{IsIndex}(i + 1, \eta, u, v)$
  $r \triangleq \textsc{choose}\ i \in \mathit{Nat} : \mathit{IsIndex}(i, \eta, u, v)$
IN
    IF $[\![\mathit{IsAFunction}(f)]\!]_{J|_{f \triangleq u}}$ THEN
     IF $\exists\, i \in \mathit{Nat} : \land\ \mathit{IsIndex}(i, \eta, u, v)$   THEN $\eta[k + 1][\![x]\!]$
                         $\land\ \neg\mathit{IsIndex}(i + 1, \eta, u, v)$
      ELSE IF $\exists\, i \in \mathit{Nat} : \mathit{IsIndex}(i, \eta, u, v)$ THEN $\eta[r][\![x]\!]$
      ELSE $\mathit{ApplyFunc}_J(u, v)$
     ELSE $\mathit{ApplyFunc}_J(u, v)$
```

$\langle 1\rangle 14.\ \forall\, r : \forall_{\text{state}}\, s : s[\![\textsc{domain}\ f]\!]_{R|_{f \triangleq r}} = s[\![\textsc{domain}\ f]\!]_{J|_{f \triangleq r}}$

$\quad\langle 2\rangle 1.$ SUFFICES: ASSUME: 1. ZF NEW r, s
$\qquad\qquad\qquad\qquad\qquad\qquad$ 2. $\mathit{IsAState}(s)$
$\qquad\qquad\qquad\quad$ PROVE: $s[\![\textsc{domain}\ f]\!]_{R|_{f \triangleq r}} = s[\![\textsc{domain}\ f]\!]_{J|_{f \triangleq r}}$

$\qquad$ BY $\forall$G.

$\quad\langle 2\rangle 2.\ \mathit{Flatten}(\mathit{Tokenize}(\text{“}\textsc{domain}\ g\text{”}), s, J|_{f \triangleq r}) = \mathit{DomainOf}(r)$

$$\begin{aligned}&\mathit{Flatten}(\mathit{Tokenize}(\text{“DOMAIN } g\text{”}), s, J|_{f \triangleq r}) = \\ &\quad \mathit{Flatten}(\langle \text{“DOMAIN ”}, \text{“}g\text{”}\rangle, s, J|_{f \triangleq r}) = \\ &\quad \mathit{DomainOf}(\mathit{Flatten}(\langle \text{“}g\text{”}\rangle, s, J|_{f \triangleq r})) = \\ &\quad \mathit{DomainOf}(r)\end{aligned}$$

⟨2⟩3. $\mathit{Flatten}(\mathit{Tokenize}(\text{“DOMAIN } g\text{”}), s, R|_{f \triangleq r}) = \mathit{DomainOf}(r)$
  Similar to PROOF of ⟨2⟩2.
⟨2⟩4. $\mathit{Flatten}(\mathit{Tokenize}(\text{“DOMAIN } g\text{”}), s, J|_{f \triangleq r}) = \mathit{Flatten}(\mathit{Tokenize}(\text{“DOMAIN } g\text{”}), s, R|_{f \triangleq r})$
  BY ⟨2⟩2, ⟨2⟩3.
⟨2⟩5. Q.E.D.
  BY ⟨2⟩4, structure of interpretations.

⟨1⟩15. $\forall\, q, r : r \in \mathit{DomainOf}(q) \Rightarrow (\mathit{ApplyFunc}_R(q, r) = \mathit{ApplyFunc}_J(q, r))$
  ⟨2⟩1. $\models \forall\, a, b, c, S : \langle a, b\rangle \notin \mathit{FirstTwo}(S) \Rightarrow \langle a, b, c\rangle \notin S$
    ⟨3⟩1. SUFFICES: ASSUME: NEW $a, b, S : \langle a, b\rangle \notin \mathit{FirstTwo}(S)$
      PROVE: $\forall\, c : \langle a, b, c\rangle \notin S$
      BY DP, ∀G.
    ⟨3⟩2. $\mathit{FirstTwo}(S) = \{\langle u, v\rangle : \exists\, c : \langle u, v, c\rangle \in S\}$
      BY ⟨1⟩3/1.
    ⟨3⟩3. $\neg \exists\, c : \langle a, b, c\rangle \in S$
      BY ⟨3⟩1, ⟨3⟩2.
    ⟨3⟩4. Q.E.D.
      BY ⟨3⟩3.
  ⟨2⟩2. ASSUME: 1. NEW S
      2. VARIABLES x, h, m
    PROVE: $\models \Box(\langle x, h\rangle \notin \mathit{FirstTwo}(S)) \Rightarrow \Box(\langle x, h, m\rangle \notin S)$
    BY ⟨2⟩1, ∀E, STL4.
  ⟨2⟩3. SUFFICES: ASSUME: ZF NEW $q, r : r \in \mathit{DomainOf}(q)$
      PROVE: $\mathit{ApplyFunc}_R(q, r) = \mathit{ApplyFunc}_J(q, r)$
    BY DP, ∀G.
  ⟨2⟩4. CASE: $\neg [\![\mathit{IsAFunction}(f)]\!]_{J|_{f \triangleq q}}$
    BY ⟨1⟩13 (DEF $\mathit{ApplyFunc}_R$).
  ⟨2⟩5. CASE: $[\![\mathit{IsAFunction}(f)]\!]_{J|_{f \triangleq q}}$
    ⟨3⟩1. $\eta \;\triangleq\; \mathit{PickLosingBeh}(q)$
    ⟨3⟩2. CASE: $\neg \exists\, i \in \mathit{Nat} : \mathit{IsIndex}(i, \eta, q, r)$
      BY ⟨1⟩13 (DEF $\mathit{ApplyFunc}_R$).
    ⟨3⟩3. ASSUME: $\exists\, i \in \mathit{Nat} : \mathit{IsIndex}(i, \eta, q, r)$
      PROVE: FALSE
      ⟨4⟩1. $\neg J|_{f \triangleq q}, \eta \models \Box(\langle x, h, m\rangle \notin \text{DOMAIN } f)$
        ⟨5⟩1. $\exists\, l \in \mathit{Nat} : r = \eta[l][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq q}}$
          ⟨6⟩1. CASE: $\neg \exists\, i \in \mathit{Nat} : \wedge\ \mathit{IsIndex}(i, \eta, q, r)$
            $\wedge\ \neg \mathit{IsIndex}(i + 1, \eta, q, r)$
            ⟨7⟩1. $l \;\triangleq\;$ CHOOSE $i \in \mathit{Nat} : \mathit{IsIndex}(i, \eta, q, r)$

```
            ⟨7⟩2. $\land\ l \in \mathit{Nat}$
                  $\land\ \mathit{IsIndex}(l, \eta, q, r)$
              BY ⟨7⟩1, ⟨3⟩3/A.
            ⟨7⟩3. $r = \eta[l][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq q}}$
              BY ⟨7⟩2, ⟨1⟩12.
            ⟨7⟩4. Q.E.D.
              BY ⟨7⟩3, ⟨7⟩2, $\exists$G WITNESS $l$.
          ⟨6⟩2. CASE: $\exists\, i \in \mathit{Nat} : \land\ \mathit{IsIndex}(i, \eta, q, r)$
                                        $\land\ \neg \mathit{IsIndex}(i+1, \eta, q, r)$
            ⟨7⟩1. $l \;\triangleq\;$ CHOOSE $\in \mathit{Nat} : \land\ \mathit{IsIndex}(i, \eta, q, r)$
                                                $\land\ \neg \mathit{IsIndex}(i+1, \eta, q, r)$
            ⟨7⟩2. $\land\ l \in \mathit{Nat}$
                  $\land\ \mathit{IsIndex}(l, \eta, q, r)$
                  $\land\ \neg \mathit{IsIndex}(l+1, \eta, q, r)$
              BY ⟨7⟩1, ⟨6⟩2/A.
            ⟨7⟩3. $\mathit{IsIndex}(l, \eta, q, r)$
              BY ⟨7⟩2, $\land$E.
            ⟨7⟩4. $r = \eta[l][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq q}}$
              BY ⟨7⟩3, ⟨1⟩12.
            ⟨7⟩5. Q.E.D.
              BY ⟨7⟩4, ⟨7⟩2, $\exists$G WITNESS $l$.
          ⟨6⟩3. Q.E.D.
            BY ⟨6⟩1, ⟨6⟩2, which are exhaustive.
        ⟨5⟩2. $\exists\, l \in \mathit{Nat} : \eta[l][\![\langle x, h, m\rangle \in \text{DOMAIN } f]\!]_{J|_{f \triangleq q}}$
          ⟨6⟩1. $r \in \mathit{DomainOf}(q)$
            BY ⟨2⟩3/A.
          ⟨6⟩2. $\exists\, l \in \mathit{Nat} : \eta[l][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq q}} \in \mathit{DomainOf}(q)$
            BY ⟨5⟩1, ⟨6⟩1.
          ⟨6⟩3. Q.E.D.
            BY ⟨6⟩2.
        ⟨5⟩3. Q.E.D.
          ⟨6⟩1. $\neg \forall\, l \in \mathit{Nat} : \eta[l][\![\langle x, h, m\rangle \notin \text{DOMAIN } f]\!]_{J|_{f \triangleq q}}$
            BY ⟨5⟩2, constant operator semantics in TLA$^+$.
          ⟨6⟩2. $\neg \forall\, l \in \mathit{Nat} : \langle \eta[l], \eta[l+1]\rangle[\![\langle x, h, m\rangle \notin \text{DOMAIN } f]\!]_{J|_{f \triangleq q}}$
            BY ⟨6⟩1, action semantics in TLA$^+$.
          ⟨6⟩3. Q.E.D.
            BY ⟨6⟩2, temporal formula semantics in TLA$^+$.
    ⟨4⟩2. $J|_{f \triangleq q}, \eta \models \Box(\langle x, h, m\rangle \notin \text{DOMAIN } f)$
        ⟨5⟩1. $\forall\, u : \Gamma(\mathit{PickLosingBeh}(u), u)$
          BY ⟨1⟩11.
        ⟨5⟩2. $\Gamma(\mathit{PickLosingBeh}(q), q)$
          BY ⟨5⟩1, $\forall$E with $u \;\triangleq\; q$.
        ⟨5⟩3. $\Gamma(\eta, q)$
          BY ⟨5⟩2, ⟨3⟩1.
```

$\langle 5\rangle 4.\ \land\ IsABeh(\eta)$
$\quad\land\ \sigma \sim_{x,h,m} \eta$
$\quad\land\ J|_{f \triangleq q}, \eta \models W(x, h, m, f)$
BY $\langle 5\rangle 3$, $\langle 1\rangle 9$.
$\langle 5\rangle 5.\ J|_{f \triangleq q}, \eta \models \lor\ \neg IsAFunction(f)$
$\quad\lor\ \land\ \neg\varphi(x, h)$
$\quad\quad\land\ m = \langle f, 0\rangle$
$\quad\quad\land\ \Box[m' = \langle f, m[2] + 1\rangle]_{\langle m,x,h\rangle}$
$\quad\quad\land\ \Box(\langle x, h\rangle \notin FirstTwo(\text{DOMAIN } f)$
BY $\langle 5\rangle 4$, $\langle 1\rangle 6/1$.
$\langle 5\rangle 6.\ J|_{f \triangleq q}, \eta \models \Box(\langle x, h\rangle \notin FirstTwo(\text{DOMAIN } f))$
BY $\langle 5\rangle 5$, $\langle 2\rangle 5$/A.
$\langle 5\rangle 7.\ J|_{f \triangleq q}, \eta \models \Box(\langle x, h\rangle \notin FirstTwo(\text{DOMAIN } f)) \Rightarrow \Box(\langle x, h, m\rangle \notin \text{DOMAIN } f)$
BY $\langle 2\rangle 2$, $\forall$E with $S \ \triangleq\ $ DOMAIN f.
$\langle 5\rangle 8.$ Q.E.D.
BY $\langle 5\rangle 6$, $\langle 5\rangle 7$, MP.
$\langle 4\rangle 3.$ Q.E.D.
BY $\langle 4\rangle 1$, $\langle 4\rangle 2$.
$\langle 3\rangle 4.$ Q.E.D.
BY $\langle 3\rangle 1$, $\langle 3\rangle 2$, which are exhaustive.
$\langle 2\rangle 6.$ Q.E.D.
BY $\langle 2\rangle 4$, $\langle 2\rangle 5$, which are exhaustive.
$\langle 1\rangle 16.$ DEFINE: Interpretation R using $Tokenize, Flatten_R, ApplyFunc_R, MakeFunction$ (so the only difference with J is replacement of the operator $ApplyFunc_J$ with $ApplyFunc_R$ and use of $ApplyFunc_R$ in the recursive definition of $Flatten_R$).
$\langle 1\rangle 17.\ IsAModel(R, \sigma)$
BY $\langle 1\rangle 16$, ASSUMPTION 3, 4, Proposition 5.
$\langle 1\rangle 18.$ SUFFICES: $R, \sigma \models F$
$\langle 2\rangle 1.\ \not\models \neg F$
$\langle 3\rangle 1.\ \land\ IsAModel(R, \sigma)$
$\quad\land\ R, \sigma \models F$
BY $\langle 1\rangle 17$, $\langle 1\rangle 18$.
$\langle 3\rangle 2.\ R, \sigma \not\models \neg F$
BY $\langle 3\rangle 1$, DEF $\models$ (semantic validity) [18, p.22].
$\langle 3\rangle 3.$ Q.E.D.
BY $\langle 3\rangle 2$, DEF $\models$.
$\langle 2\rangle 2.\ \not\models G$
$\langle 3\rangle 1.\ \not\models \neg F$
BY $\langle 2\rangle 1$.
$\langle 3\rangle 2.\ G \equiv \neg F$
$\langle 4\rangle 1.\ G = \exists f : \land\ IsAFunction(f)$
$\quad\land\ \forall x, h : \psi(x, h, f) \Rightarrow \varphi(x, h)$
BY ASSUMPTION 11.
$\langle 4\rangle 2.\ F = \forall f : IsAFunction(f) \Rightarrow \exists x, h : \psi(x, h, f) \land \neg\varphi(x, h)$

BY $\langle 1\rangle 1$.

$\langle 4\rangle 3$. Q.E.D.

BY $\langle 4\rangle 1$, $\langle 4\rangle 2$.

$\langle 3\rangle 3$. Q.E.D.

BY $\langle 3\rangle 1$, $\langle 3\rangle 2$.

$\langle 2\rangle 3$. $\not\models G \Rightarrow \nvdash G$

$\langle 3\rangle 1$. TLA$^+$ is sound

BY ASSUMPTION 1.

$\langle 3\rangle 2$. $(\vdash G) \Rightarrow (\models G)$

BY $\langle 3\rangle 1$, DEF soundness.

$\langle 3\rangle 3$. Q.E.D.

BY $\langle 3\rangle 2$, contrapositive.

$\langle 2\rangle 4$. Q.E.D.

BY $\langle 2\rangle 2$, $\langle 2\rangle 3$, MP.

$\langle 1\rangle 19$. SUFFICES: ASSUME: ZF NEW p

PROVE: $R|_{f \triangleq p}, \sigma \models \exists\, x, h : \begin{array}{l} \lor \neg \mathit{IsAFunction}(f) \\ \lor \land \psi(x, h, f) \\ \quad \land \neg\varphi(x, h) \end{array}$

BY $\langle 1\rangle 1$ (DEF F), DEF $\forall$.

$\langle 1\rangle 20$. $\tau \triangleq \mathit{PickLosingBeh}(p)$

$\langle 1\rangle 21$. $\Gamma(\tau, p)$

BY $\langle 1\rangle 11$, $\langle 1\rangle 20$, $\forall$E with $u \triangleq p$.

$\langle 1\rangle 22$. $J|_{f \triangleq p}, \tau \models W(m, x, h, f)$

BY $\langle 1\rangle 21$, $\langle 1\rangle 9$.

$\langle 1\rangle 23$. ASSUME: $R|_{f \triangleq p}, \tau \models \mathit{IsAFunction}(f)$

PROVE: $J|_{f \triangleq p}, \tau \models \mathit{IsAFunction}(f)$

$\langle 2\rangle 1$. $\mathit{Flatten}(\mathit{Tokenize}(\text{“}[x \in \textsc{domain}\ f \mapsto f[x]] = f\text{”}, \tau[0], R|_{f \triangleq p}))$

BY $\langle 1\rangle 23$/A, $\langle 1\rangle 16$.

$\langle 2\rangle 2$. $\mathit{Flatten}(\langle \text{“}f\text{”} \rangle, \tau[0], R|_{f \triangleq p})$
$= \mathit{Flatten}(\mathit{Tokenize}(\text{“}[x \in \textsc{domain}\ f \mapsto f[x]]\text{”}), \tau[0], R|_{f \triangleq p})$

BY $\langle 2\rangle 1$.

$\langle 2\rangle 3$. $\mathit{Flatten}(\langle \text{“}f\text{”} \rangle, \tau[0], R|_{f \triangleq p}) = \mathit{Flatten}(\langle \text{“}f\text{”} \rangle, \tau[0], J|_{f \triangleq p})$

$\langle 3\rangle 1$. $\mathit{Flatten}(\langle \text{“}f\text{”} \rangle, \tau[0], R|_{f \triangleq p}) = p$

$\langle 3\rangle 2$. $\mathit{Flatten}(\langle \text{“}f\text{”} \rangle, \tau[0], J|_{f \triangleq p}) = p$

$\langle 3\rangle 3$. Q.E.D.

BY $\langle 3\rangle 1$, $\langle 3\rangle 2$.

$\langle 2\rangle 4$. $\mathit{Flatten}(\mathit{Tokenize}(\text{“}[x \in \textsc{domain}\ f \mapsto f[x]]\text{”}), \tau[0], R|_{f \triangleq p})$
$= \mathit{Flatten}(\mathit{Tokenize}(\text{“}[x \in \textsc{domain}\ f \mapsto f[x]]\text{”}), \tau[0], J|_{f \triangleq p})$

⟨3⟩1.

$$\begin{aligned}&\mathit{Flatten}(\mathit{Tokenize}(\text{“}[x \in \textsc{domain}\ f \mapsto f[x]]\text{”}), \tau[0], R|_{f \triangleq p})\\ &= \mathit{MakeFunction}(\{\\ &\quad \langle r, \mathit{ApplyFunc}_R(\\ &\qquad \mathit{Flatten}(\langle \text{“}f\text{”} \rangle, \tau[0], R_{f \triangleq p, x \triangleq r}),\\ &\qquad \mathit{Flatten}(\langle \text{“}x\text{”} \rangle, \tau[0], R_{f \triangleq p, x \triangleq r})\\ &\quad)\rangle\ :\\ &\quad r \in \mathit{Flatten}(\langle \text{“}\textsc{domain}\ \text{”}, \text{“}f\text{”} \rangle, \tau[0], R|_{f \triangleq p}) \quad \})\end{aligned}$$

⟨3⟩2.

$$\begin{aligned}@ =&\mathit{MakeFunction}(\{\\ &\quad \langle r, \mathit{ApplyFunc}_R(p, r)\rangle\ :\\ &\quad r \in \mathit{DomainOf}\big(\mathit{Flatten}(\langle \text{“}f\text{”} \rangle, \tau[0], R|_{f \triangleq p})\big) \quad \})\end{aligned}$$

⟨3⟩3. $@ = \mathit{MakeFunction}(\{\langle r, \mathit{ApplyFunc}_R(p, r)\rangle\ :\ r \in \mathit{DomainOf}(p)\})$
⟨3⟩4. $\forall\, r \in \mathit{DomainOf}(p)\ :\ \mathit{ApplyFunc}_R(p, r) = \mathit{ApplyFunc}_J(p, r)$
BY ⟨1⟩15, ∀E with $q \ \triangleq\ p$.
⟨3⟩5. $\mathit{MakeFunction}(\{\langle r, \mathit{ApplyFunc}_R(p, r)\rangle\ :\ r \in \mathit{DomainOf}(p)$
$= \mathit{MakeFunction}(\{\langle r, \mathit{ApplyFunc}_J(p, r)\rangle\ :\ r \in \mathit{DomainOf}(p)$
BY ⟨3⟩3, ⟨3⟩4.
⟨3⟩6.

$$\begin{aligned}@ = &\mathit{MakeFunction}(\{\\ &\quad \langle r, \mathit{ApplyFunc}_J(\\ &\qquad \mathit{Flatten}(\langle \text{“}f\text{”} \rangle, \tau[0], J_{f \triangleq p, x \triangleq r}),\\ &\qquad \mathit{Flatten}(\langle \text{“}x\text{”} \rangle, \tau[0], J_{f \triangleq p, x \triangleq r})\\ &\quad)\rangle\ :\\ &\quad r \in \mathit{Flatten}(\langle \text{“}\textsc{domain}\ \text{”}, \text{“}f\text{”} \rangle, \tau[0], J|_{f \triangleq p}) \quad \})\end{aligned}$$

⟨3⟩7. $@ = \mathit{Flatten}(\mathit{Tokenize}(\text{“}[x \in \textsc{domain}\ f \mapsto f[x]]\text{”}), \tau[0], J|_{f \triangleq p})$
⟨3⟩8. Q.E.D.
BY ⟨3⟩1, ⟨3⟩2, ⟨3⟩3, ⟨3⟩5, ⟨3⟩6, ⟨3⟩7.
⟨2⟩5. $\mathit{Flatten}(\langle \text{“}f\text{”} \rangle, \tau[0], J|_{f \triangleq p})$
$= \mathit{Flatten}(\mathit{Tokenize}(\text{“}[x \in \textsc{domain}\ f \mapsto f[x]]\text{”}), \tau[0], J|_{f \triangleq p})$
BY ⟨2⟩2, ⟨2⟩3, ⟨2⟩4.
⟨2⟩6. $\mathit{Flatten}(\mathit{Tokenize}(\text{“}[x \in \textsc{domain}\ f \mapsto f[x]]\text{”}), \tau[0], J|_{f \triangleq p})$
BY ⟨2⟩5.
⟨2⟩7. Q.E.D.
BY ⟨2⟩6.
⟨1⟩24. ASSUME: 1. ZF NEW $n, k \in \mathit{Nat}$
2. $J|_{f \triangleq p}, \tau \models \land\ m = \langle f, 0\rangle$
$\land\ \Box[m' = \langle f, m[2] + 1\rangle]_{\langle x, h, m\rangle}$

3. $\land\ \tau[n] =_{x,h,m} \tau[k]$
$\land\ \tau[n] \neq_{x,h,m} \tau[n+1]$
$\land\ \tau[k] \neq_{x,h,m} \tau[k+1]$

PROVE: $n = k$

⟨2⟩1. SUFFICES: ASSUME: $n \neq k$
PROVE: FALSE

⟨3⟩1. $n, k \in \mathit{Nat}$
BY ⟨1⟩24/A1.

⟨3⟩2. $(n = k) \lor (n \neq k)$
BY ⟨3⟩1, *Naturals*.

⟨3⟩3. Q.E.D.
BY ⟨3⟩2, ⟨2⟩1.

⟨2⟩2. CASE: $n < k$

⟨3⟩1. $\tau \models \land\ m[2] \in \mathit{Nat}$
$\land\ \Box[(m[2])' = m[2] + 1]_{\langle m,x,h \rangle}$

⟨4⟩1. $\tau \models m[2] \in \mathit{Nat}$

⟨5⟩1. $\models (m = \langle f, 0 \rangle) \Rightarrow (m[2] = 0)$

⟨5⟩2. $\tau \models (m = \langle f, 0 \rangle)$
BY ⟨1⟩24/A2, $\land$E.

⟨5⟩3. $\tau \models m[2] = 0$
BY ⟨5⟩1, ⟨5⟩2, MP.

⟨5⟩4. Q.E.D.
BY ⟨5⟩3, *Naturals*.

⟨4⟩2. $\tau \models \Box[(m[2])' = m[2] + 1]_{\langle m,x,h \rangle}$

⟨5⟩1. $\models [m' = \langle f, m[2] + 1 \rangle]_{m,x,h\rangle} \Rightarrow [(m[2])' = m[2] + 1]_{\langle m,x,h \rangle}$

⟨5⟩2. $\models \Box[m' = \langle f, m[2] + 1 \rangle]_{\langle m,x,h \rangle} \Rightarrow \Box[(m[2])' = m[2] + 1]_{\langle m,x,h \rangle}$
BY ⟨5⟩1, TLA2.

⟨5⟩3. $\tau \models \Box[m' = \langle f, m[2] + 1 \rangle]_{\langle m,x,h \rangle} \Rightarrow \Box[(m[2])' = m[2] + 1]_{\langle m,x,h \rangle}$
BY ⟨5⟩2, DEF $\models$, ⟨1⟩21, ⟨1⟩9 (*IsABeh*(τ)).

⟨5⟩4. Q.E.D.
BY ⟨1⟩24/A2 $\land$E, ⟨5⟩3, MP.

⟨4⟩3. Q.E.D.
BY ⟨4⟩1, ⟨4⟩2, $\land$I.

⟨3⟩2. $\tau \models \Box(m[2] \in \mathit{Nat})$

⟨4⟩1. $\begin{pmatrix} \land\ m[2] \in \mathit{Nat} \\ \land\ [(m[2])' = m[2] + 1]_{\langle m,x,h \rangle} \end{pmatrix} \Rightarrow (m[2] \in \mathit{Nat})'$

PROOF:

$$\begin{array}{l} \land\ m[2] \in Nat \\ \land\ [(m[2])' = m[2]+1]_{\langle m,x,h\rangle} \end{array} = \begin{array}{l} \land\ m[2] \in Nat \\ \land \begin{array}{l} \lor\ (m[2])' = m[2]+1 \\ \lor\ \langle m,x,h\rangle' = \langle m,x,h\rangle \end{array} \end{array}$$

$$= \begin{array}{l} \lor \begin{array}{l} \land\ m[2] \in Nat \\ \land\ m' = m \end{array} \\ \lor \begin{array}{l} \land\ m[2] \in Nat \\ \land\ (m[2])' = m[2]+1 \end{array} \end{array}$$

$$\Rightarrow \begin{array}{l} \lor\ (m[2] \in Nat)' \\ \lor \begin{array}{l} \land\ (m[2]+1) \in Nat \\ \land\ (m[2])' = (m[2]+1) \end{array} \end{array} \Rightarrow (m[2] \in Nat)'$$

⟨4⟩2. $\models \left(\begin{array}{l} \land\ m[2] \in Nat \\ \land\ \Box[(m[2])' = m[2]+1]_{\langle m,x,h\rangle} \end{array}\right) \Rightarrow \Box(m[2] \in Nat)$

BY ⟨3⟩1, INV1 with $P \triangleq m[2] \in Nat$.

⟨4⟩3. $\tau \models \begin{array}{l} \land\ m[2] \in Nat \\ \land\ \Box[(m[2])' = m[2]+1]_{\langle m,x,h\rangle} \end{array} \Rightarrow \Box(m[2] \in Nat)$

BY ⟨4⟩2, DEF $\models$, ⟨1⟩21, ⟨1⟩9 ($IsABeh(\tau)$).

⟨4⟩4. Q.E.D.

BY ⟨3⟩1, ⟨4⟩3, MP.

⟨3⟩3. $\tau \models \Box\left[\begin{array}{l} \land\ m[2] \in Nat \\ \land\ (m[2])' = m[2]+1 \end{array}\right]_{\langle m,x,h\rangle}$

BY ⟨3⟩1, $\land$E, ⟨3⟩2, INV2, MP.

⟨3⟩4. $\langle \tau[n], \tau[n+1]\rangle \left[\!\left[\begin{array}{l} \land\ m[2] \in Nat \\ \land\ (m[2])' = m[2]+1 \end{array}\right]\!\right]$

⟨4⟩1. $\langle \tau[n], \tau[n+1]\rangle \left[\!\left[\left[\begin{array}{l} \land\ m[2] \in Nat \\ \land\ (m[2])' = m[2]+1 \end{array}\right]_{\langle m,x,h\rangle}\right]\!\right]$

BY ⟨3⟩3, TLA$^+$ semantics, ⟨1⟩24/A1, $\forall$E.

⟨4⟩2. $\tau[n] \neq_{x,h,m} \tau[n+1]$

BY ⟨1⟩24/A3, $\land$E.

⟨4⟩3. Q.E.D.

BY ⟨4⟩1, ⟨4⟩2.

⟨3⟩5. $\langle \tau[n], \tau[n+1]\rangle [\![(m[2])' \leq m[2]]\!]_{J|_{f \triangleq p}}$

⟨4⟩1. $\forall\, i, j \in Nat : (i \leq j) \Rightarrow \langle \tau[i], \tau[j]\rangle [\![m[2] \leq (m[2])']\!]_{J|_{f \triangleq p}}$

BY ⟨3⟩3, (finitary) induction.

⟨4⟩2. $(n+1) \leq k$

⟨5⟩1. $n, k \in Nat$

BY ⟨1⟩24/A1.

⟨5⟩2. $n < k$

BY ⟨2⟩2.

⟨5⟩3. Q.E.D.

BY ⟨5⟩1, ⟨5⟩2, *Naturals*.

⟨4⟩3. $(n+1 \leq k) \Rightarrow \langle \tau[n], \tau[k]\rangle [\![m[2] \leq (m[2])']\!]_{J|_{f \triangleq p}}$

BY ⟨3⟩1, ⟨1⟩24/A1, $\forall$E with $i \triangleq n+1, j \triangleq k$.

$\langle 4\rangle 4.$ $\langle \tau[n+1], \tau[k]\rangle [\![m[2] \leq (m[2])']\!]_{J|_{f \triangleq p}}$
BY $\langle 4\rangle 2$, $\langle 4\rangle 3$, MP.
$\langle 4\rangle 5.$ $\tau[k] =_{x,h,m} \tau[n]$
BY $\langle 1\rangle 24$/A3.
$\langle 4\rangle 6.$ $\tau[k][\![m]\!] = \tau[n][\![m]\!]$
BY $\langle 4\rangle 5$, DEF $=_{x,h,m}$ (Eq. (4) on page 9).
$\langle 4\rangle 7.$ $\langle \tau[n+1], \tau[n]\rangle [\![m[2] \leq (m[2])']\!]_{J|_{f \triangleq p}}$
BY $\langle 4\rangle 4$, $\langle 4\rangle 6$, m is the only variable symbol that appears in the expression "$m[2] \leq (m[2])'$".
$\langle 4\rangle 8.$ Q.E.D.
BY $\langle 4\rangle 7$, TLA$^+$ action semantics.
$\langle 3\rangle 6.$ $\langle \tau[n], \tau[n+1]\rangle [\![\begin{array}{l} \wedge\ m[2] \in Nat \\ \wedge\ (m[2])' = m[2]+1 \\ \wedge\ (m[2])' \leq m[2] \end{array}]\!]_{J|_{f \triangleq p}}$
BY $\langle 3\rangle 4$, $\langle 3\rangle 5$.
$\langle 3\rangle 7.$ $\langle \tau[n], \tau[n+1]\rangle [\![\begin{array}{l} \wedge\ m[2] \in Nat \\ \wedge\ m[2]+1 \leq m[2] \end{array}]\!]_{J|_{f \triangleq p}}$
BY $\langle 3\rangle 6$.
$\langle 3\rangle 8.$ $\langle \tau[n], \tau[n+1]\rangle [\![\text{FALSE}]\!]_{J|_{f \triangleq p}}$
BY $\langle 3\rangle 7$, *Naturals*.
$\langle 3\rangle 9.$ $\tau[n][\![\text{FALSE}]\!]_{J|_{f \triangleq p}}$
BY $\langle 3\rangle 8$, TLA$^+$ action semantics.
$\langle 3\rangle 10.$ Q.E.D.
BY $\langle 3\rangle 9$, DEF $[\![]\!]$ for an interpretation, ASSUMPTION 3, 4 ($J|_{f \triangleq p}$ is a model).
$\langle 2\rangle 3.$ CASE: $n > k$
PROOF: Similar to that of $\langle 2\rangle 2$.
$\langle 2\rangle 4.$ Q.E.D.
$\langle 3\rangle 1.$ $\begin{array}{l} \vee\ n < k \\ \vee\ n > k \end{array} \equiv \begin{array}{l} \wedge\ n, k \in Nat \\ \wedge\ n \neq k \end{array}$
BY *Naturals*.
$\langle 3\rangle 2.$ $(n < k) \vee (n > k)$
BY $\langle 3\rangle 1$, $\langle 1\rangle 24$/A1, $\langle 2\rangle 1$/A, MP.
$\langle 3\rangle 3.$ Q.E.D.
BY $\langle 3\rangle 2$, the cases $\langle 2\rangle 2$, $\langle 2\rangle 3$ are exhaustive.
$\langle 1\rangle 25.$ $R|_{f \triangleq p}, \tau \models \begin{array}{l} \vee\ \neg IsAFunction(f) \\ \vee\ \wedge\ \Box[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle} \\ \quad\ \wedge\ \neg\varphi(x,h) \end{array}$
$\langle 2\rangle 1.$ $R|_{f \triangleq p}, \tau \models \begin{array}{l} \vee\ \neg IsAFunction(f) \\ \vee\ \wedge\ \neg\varphi(x,h) \\ \quad\ \wedge\ \Box(\langle x, h\rangle \notin FirstTwo(\text{DOMAIN } f)) \\ \quad\ \wedge\ m = \langle f, 0\rangle \\ \quad\ \wedge\ \Box[m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle} \end{array}$
$\langle 3\rangle 1.$ SUFFICES: ASSUME: $[\![IsAFunction(f)]\!]_{R|_{f \triangleq p}}$

PROVE: $R|_{f \triangleq p}, \tau \models \neg\varphi(x, h)$

BY DP.

$\langle 3\rangle 2.$ $J|_{f \triangleq p}, \tau \models \mathit{IsAFunction}(f)$

BY $\langle 1\rangle 23$.

$\langle 3\rangle 3.$ $J|_{f \triangleq p}, \tau \models W(m, x, h, f)$

BY $\langle 1\rangle 22$.

$\langle 3\rangle 4.$ $J|_{f \triangleq p}, \tau \models \vee \neg \mathit{IsAFunction}(f)$
$\quad \vee \wedge \neg\varphi(x, h)$
$\quad\quad \wedge \Box(\langle x, h\rangle \notin \mathit{FirstTwo}(\text{DOMAIN } f))$
$\quad\quad \wedge m = \langle f, 0\rangle$
$\quad\quad \wedge \Box[m' = \langle f, m[2] + 1\rangle]_{\langle m,x,h\rangle}$

BY $\langle 3\rangle 4$, $\langle 1\rangle 6/1$.

$\langle 3\rangle 5.$ $J|_{f \triangleq p}, \tau \models \vee \neg \mathit{IsAFunction}(f)$
$\quad \vee \neg\varphi(x, h)$

BY $\langle 3\rangle 4$, $\wedge$E.

$\langle 3\rangle 6.$ $J|_{f \triangleq p}, \tau \models \neg\varphi(x, h)$

BY $\langle 3\rangle 2$, $\langle 3\rangle 5$, MP.

$\langle 3\rangle 7.$ Q.E.D.

BY $\langle 3\rangle 6$, $\langle 1\rangle 16$, ASSUMPTION 7.

$\langle 2\rangle 2.$ $R|_{f \triangleq p}, \tau \models \vee \neg \mathit{IsAFunction}(f)$
$\quad \vee \Box[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}$

$\langle 3\rangle 1.$ SUFFICES: ASSUME: $R|_{f \triangleq p}, \tau \models \mathit{IsAFunction}(f)$
PROVE: $R|_{f \triangleq p} \models \Box[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}$

BY DP, TLA$^+$ semantics.

$\langle 3\rangle 2.$ SUFFICES: ASSUME: ZF NEW $k \in \mathit{Nat}$
PROVE: $\langle \tau[k], \tau[k+1]\rangle [\![[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}]\!]_{R|_{f \triangleq p}}$

$\langle 4\rangle 1.$ $\forall k \in \mathit{Nat} : \langle \tau[k], \tau[k+1]\rangle [\![[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}]\!]_{R|_{f \triangleq p}}$

BY $\langle 3\rangle 2$.

$\langle 4\rangle 2.$ Q.E.D.

BY $\langle 4\rangle 1$, TLA$^+$ semantics.

$\langle 3\rangle 3.$ SUFFICES: ASSUME: $\langle \tau[k], \tau[k+1]\rangle [\![\neg\text{UNCHANGED } \langle x, h, m\rangle]\!]_{R|_{f \triangleq p}}$
PROVE: $\langle \tau[k], \tau[k+1]\rangle [\![x' = f[\langle x, h, m\rangle]]\!]_{R|_{f \triangleq p}}$

$\langle 4\rangle 1.$ $\vee \neg\langle \tau[k], \tau[k+1]\rangle [\![\neg\text{UNCHANGED } \langle x, h, m\rangle]\!]_{R|_{f \triangleq p}}$
$\quad \vee \langle \tau[k], \tau[k+1]\rangle [\![x' = f[\langle x, h, m\rangle]]\!]_{R|_{f \triangleq p}}$

BY $\langle 3\rangle 3$.

$\langle 4\rangle 2.$ $\vee \langle \tau[k], \tau[k+1]\rangle [\![\text{UNCHANGED } \langle x, h, m\rangle]\!]_{R|_{f \triangleq p}}$
$\quad \vee \langle \tau[k], \tau[k+1]\rangle [\![x' = f[\langle x, h, m\rangle]]\!]_{R|_{f \triangleq p}}$

BY $\langle 4\rangle 1$, TLA$^+$ semantics.

$\langle 4\rangle 3.$ $\langle \tau[k], \tau[k+1]\rangle [\![(x' = f[\langle x, h, m\rangle]) \vee \text{UNCHANGED } \langle x, h, m\rangle]\!]_{R|_{f \triangleq p}}$

BY $\langle 4\rangle 2$, TLA$^+$ semantics.

$\langle 4\rangle 4.$ Q.E.D.

BY $\langle 4\rangle 3$, DEF $[A]_v$.

$\langle 3\rangle 4.$ Q.E.D.

$\langle 4 \rangle 1.$ $\langle \tau[k], \tau[k+1] \rangle [\![x' = f[\langle x, h, m \rangle]]\!]_{R|_{f \triangleq p}}$
$\equiv \big(\tau[k+1] [\![x]\!] = \mathit{ApplyFunc}_R(p, \tau[k] [\![\langle x, h, m \rangle]\!]_{R|_{f \triangleq p}}) \big)$

$\langle 5 \rangle 1.$ $\langle \tau[k], \tau[k+1] \rangle [\![x' = f[\langle x, h, m \rangle]]\!]_{R|_{f \triangleq p}}$
$\equiv \big(\tau[k+1] [\![x]\!]_{R|_{f \triangleq p}} = \tau[k] [\![f[\langle x, h, m \rangle]]\!]_{R|_{f \triangleq p}} \big)$

BY TLA$^+$ semantics.

$\langle 5 \rangle 2.$ $\tau[k+1] [\![x]\!]_{R|_{f \triangleq p}} = \tau[k+1] [\![x]\!]$

BY TLA$^+$ semantics: $\tau[k+1] [\![x]\!]_{R|_{f \triangleq p}} = \tau[k+1][\text{“}x\text{”}]$. We use $\tau[k+1] [\![x]\!]$ to denote $\tau[k+1][\text{“}x\text{”}]$, emphasizing that it is the same in $R|_{f \triangleq p}$ and $J|_{f \triangleq p}$.

$\langle 5 \rangle 3.$ $\tau[k] [\![f[\langle x, h, m \rangle]]\!]_{R|_{f \triangleq p}} = \mathit{ApplyFunc}_R(p, \tau[k] [\![\langle x, h, m \rangle]\!]_{R|_{f \triangleq p}})$

$$\begin{aligned}
&\tau[k] [\![f[\langle x, h, m \rangle]]\!]_{R|_{f \triangleq p}} \\
&= \mathit{Flatten}(\mathit{Tokenize}(\text{“}f[\langle x, h, m \rangle]\text{”}), \tau[k], R|_{f \triangleq p}) \\
&= \mathit{ApplyFunc}_R\big(\\
&\qquad \mathit{Flatten}(\langle \text{“}f\text{”} \rangle, \tau[k], R|_{f \triangleq p}), \\
&\qquad \mathit{Flatten}(\mathit{Tokenize}(\text{“}\langle x, h, m \rangle\text{”}), \tau[k], R|_{f \triangleq p}) \\
&= \mathit{ApplyFunc}_R(p,\ \tau[k] [\![\langle x, h, m \rangle]\!]_{R|_{f \triangleq p}})\big)
\end{aligned}$$

$\langle 5 \rangle 4.$ Q.E.D.

BY $\langle 5 \rangle 1$, $\langle 5 \rangle 2$, $\langle 5 \rangle 3$.

$\langle 4 \rangle 2.$ $\tau[k] [\![\langle x, h, m \rangle]\!]_{R|_{f \triangleq p}} = \tau[k] [\![\langle x, h, m \rangle]\!]_{J|_{f \triangleq p}}$

$\langle 5 \rangle 1.$ $e(i, a, b, c) \ \triangleq\ \text{IF } i = 1 \text{ THEN } a \text{ ELSE IF } i = 2 \text{ THEN } b \text{ ELSE } c$

$\langle 5 \rangle 2.$ $\tau[k] [\![\langle x, h, m \rangle]\!]_{R|_{f \triangleq p}}$
$= \mathit{MakeFunction}(\{\langle 1, \tau[k][\text{“}x\text{”}] \rangle, \langle 2, \tau[k][\text{“}h\text{”}] \rangle, \langle 3, \tau[m][\text{“}m\text{”}] \rangle\})$

PROOF: (See also Remark 8 on page 33.)

$\tau[k][\![\langle x, h, m\rangle]\!]_{R|_{f \triangleq p}}$
$= \tau[k][\![[i \in \{1,2,3\} \mapsto e(i,x,h,m)]]\!]_{R|_{f \triangleq p}}$
$= \mathit{Flatten}(\mathit{Tokenize}(\text{“}[i \in \{1,2,3\} \mapsto e(i,x,h,m)]\text{”}), \tau[k], R|_{f \triangleq p})$
$= \mathit{MakeFunction}(\{\langle r, \mathit{Flatten}(\mathit{Tokenize}(\text{“}e(i,x,h,m)\text{”}), \tau[k], R|_{f \triangleq p, i \triangleq r})\rangle :$
$r \in \mathit{Flatten}(\mathit{Tokenize}(\text{“}\{1,2,3\}\text{”}), \tau[k], R|_{f \triangleq p})\})$
$= \mathit{MakeFunction}(\{\langle r, \mathit{Flatten}(\mathit{Tokenize}(\text{“}e(i,x,h,m)\text{”}), \tau[k], R|_{f \triangleq p, i \triangleq r})\rangle :$
$r \in \{1,2,3\}$
$\})$
$= \mathit{MakeFunction}(\{$
$\langle r, \text{IF } r = 1 \text{ THEN } \tau[k][\text{“}x\text{”}]$
$\text{ELSE IF } r = 2 \text{ THEN } \tau[k][\text{“}h\text{”}]$
$\text{ELSE } \tau[k][\text{“}m\text{”}]\rangle,$
$r \in \{1,2,3\}$
$\})$
$= \mathit{MakeFunction}(\{\langle 1, \tau[k][\text{“}x\text{”}]\rangle, \langle 2, \tau[k][\text{“}h\text{”}]\rangle, \langle 3, \tau[k][\text{“}m\text{”}]\rangle\})$

⟨5⟩3. $\tau[k][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq p}}$
$= \mathit{MakeFunction}(\{\langle 1, \tau[k][\text{“}x\text{”}]\rangle, \langle 2, \tau[k][\text{“}h\text{”}]\rangle, \langle 3, \tau[m][\text{“}m\text{”}]\rangle\})$
PROOF: Similar to that of ⟨5⟩2.
⟨5⟩4. Q.E.D.
BY ⟨5⟩2, ⟨5⟩3.
⟨4⟩3. $\langle \tau[k], \tau[k+1]\rangle[\![x' = f[\langle x, h, m\rangle]]\!]_{R|_{f \triangleq p}}$
$\equiv \big(\tau[k+1][\![x]\!] = \mathit{ApplyFunc}_R(p, \tau[k][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq p}})\big)$
BY ⟨4⟩1, ⟨4⟩2.
⟨4⟩4. $\mathit{ApplyFunc}_R(p, \tau[k][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq p}}) = \tau[k+1][\![x]\!]$
⟨5⟩1. $J|_{f \triangleq p}, \tau \models \mathit{IsAFunction}(f)$
BY ⟨3⟩1/A, ⟨1⟩23.
⟨5⟩2. $\langle \tau[k], \tau[k+1]\rangle[\![\neg \text{UNCHANGED } \langle x, h, m\rangle]\!]_{J|_{f \triangleq p}}$
⟨6⟩1. $\langle \tau[k], \tau[k+1]\rangle[\![\neg \text{UNCHANGED } \langle x, h, m\rangle]\!]_{R|_{f \triangleq p}}$
BY ⟨3⟩3/A.
⟨6⟩2. $\lor\ \tau[k][\text{“}x\text{”}] \neq \tau[k+1][\text{“}x\text{”}]$
$\lor\ \tau[k][\text{“}h\text{”}] \neq \tau[k+1][\text{“}h\text{”}]$
$\lor\ \tau[k][\text{“}m\text{”}] \neq \tau[k+1][\text{“}m\text{”}]$
BY ⟨6⟩1, x, h, m not declared as CONSTANTS in $R|_{f \triangleq p}$ ⟨1⟩9, ⟨1⟩10, ⟨1⟩20, ⟨1⟩16.
⟨6⟩3. Q.E.D.
BY ⟨6⟩2, x, h, m not declared as CONSTANTS in $J|_{f \triangleq p}$ ⟨1⟩9, ⟨1⟩10, ⟨1⟩20.

$\langle 5\rangle 3.\ v \triangleq \tau[k][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq p}}$

$\langle 5\rangle 4.$

$$\begin{array}{l}
\mathit{ApplyFunc}_R(p, v) \\
= \text{LET} \\
\quad \eta \triangleq \mathit{PickLosingBeh}(p) \\
\quad j \triangleq \text{CHOOSE } i \in \mathit{Nat} : \land \mathit{IsIndex}(i, \eta, p, v) \\
\quad\qquad\qquad\qquad\qquad\quad \land \neg \mathit{IsIndex}(i+1, \eta, v) \\
\quad r \triangleq \text{CHOOSE } i \in \mathit{Nat} : \mathit{IsIndex}(i, \eta, p, v) \\
\text{IN} \\
\quad \text{IF } [\![\mathit{IsAFunction}(f)]\!]_{J|_{f \triangleq p}} \text{ THEN} \\
\qquad \text{IF } \exists\, i \in \mathit{Nat} : \land \mathit{IsIndex}(i, \eta, p, v) \\
\qquad\qquad\qquad\qquad \land \neg \mathit{IsIndex}(i+1, \eta, p, v) \\
\qquad\ \text{THEN } \eta[j+1][\![x]\!] \\
\qquad\ \text{ELSE IF } \exists\, i \in \mathit{Nat} : \mathit{IsIndex}(i, \eta, p, v) \\
\qquad\ \text{THEN } \eta[r][\![x]\!] \text{ ELSE } \mathit{ApplyFunc}_J(p, v) \\
\quad\ \text{ELSE } \mathit{ApplyFunc}_J(p, v)
\end{array}$$

BY $\langle 1\rangle 13$.

$\langle 5\rangle 5.\ \eta = \tau$

BY $\langle 5\rangle 4$, $\langle 1\rangle 20$.

$\langle 5\rangle 6.$

$$\begin{array}{l}
\mathit{ApplyFunc}_R(p, v) \\
= \text{LET} \\
\quad j \triangleq \text{CHOOSE } i \in \mathit{Nat} : \land \mathit{IsIndex}(i, \tau, p, v) \\
\quad\qquad\qquad\qquad\qquad\quad \land \neg \mathit{IsIndex}(i+1, \tau, v) \\
\quad r \triangleq \text{CHOOSE } i \in \mathit{Nat} : \mathit{IsIndex}(i, \tau, p, v) \\
\text{IN} \\
\quad \text{IF } \exists\, i\mathit{Nat} : \land \mathit{IsIndex}(i, \tau, p, v) \\
\qquad\qquad\quad \land \neg \mathit{IsIndex}(i+1, \tau, p, v) \\
\quad \text{THEN } \tau[j+1][\![x]\!] \\
\quad \text{ELSE IF } \exists\, i \in \mathit{Nat} : \mathit{IsIndex}(i, \tau, p, v) \text{ THEN } \tau[r][\![x]\!] \\
\quad \text{ELSE } \mathit{ApplyFunc}_J(p, v)
\end{array}$$

BY $\langle 5\rangle 4$, $\langle 5\rangle 1$, $\langle 5\rangle 5$.

$\langle 5\rangle 7.\ \exists\, i \in \mathit{Nat} : \land \mathit{IsIndex}(i, \tau, p, v)$
$\qquad\qquad\qquad\quad \land \neg \mathit{IsIndex}(i+1, \tau, p, v)$

$\langle 6\rangle 1.$ SUFFICES: $\exists\, i \in \mathit{Nat} : \land\, \tau[k][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq p}} = \tau[i][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq p}}$
$\qquad\qquad\qquad\qquad\qquad \land\, \tau[k][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq p}} \neq \tau[i+1][\![\langle x, h, m\rangle]\!]_{J|_{f \triangleq p}}$

BY $\langle 1\rangle 12$.

$\langle 6\rangle 2.$ $\tau[k][\![\langle x,h,m\rangle]\!]_{J|_{f\triangleq p}} = \tau[k][\![\langle x,h,m\rangle]\!]_{J|_{f\triangleq p}}$
BY ZF equality reflexivity axiom.

$\langle 6\rangle 3.$ $\langle \tau[k], \tau[k+1]\rangle[\![\neg\text{UNCHANGED } \langle x,h,m\rangle]\!]_{J|_{f\triangleq p}}$
BY $\langle 5\rangle 2$.

$\langle 6\rangle 4.$ $\tau[k][\![\langle x,h,m\rangle]\!]_{J|_{f\triangleq p}} \neq \tau[i+1][\![\langle x,h,m\rangle]\!]_{J|_{f\triangleq p}}$
BY $\langle 6\rangle 3$.

$\langle 6\rangle 5.$ Q.E.D.
BY $\langle 6\rangle 2$, $\langle 6\rangle 4$, $\wedge$I, $\exists$G with WITNESS $i \triangleq k$, $\langle 3\rangle 2$/A.

$\langle 5\rangle 8.$ $\mathit{ApplyFunc}_R(p,v) = \tau[1+\text{CHOOSE } i \in \mathit{Nat} : \wedge\ \mathit{IsIndex}(i,\tau,p,v) \quad][\![x]\!]$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \wedge\ \neg \mathit{IsIndex}(i+1,\tau,pv)$
BY $\langle 5\rangle 6$, $\langle 5\rangle 7$.

$\langle 5\rangle 9.$ SUFFICES: $k = \text{CHOOSE } i \in \mathit{Nat} : \wedge\ \mathit{IsIndex}(i,\tau,p,v)$
$\qquad\qquad\qquad\qquad\qquad\qquad \wedge\ \neg \mathit{IsIndex}(i+1,\tau,p,v)$
BY substitution of k for CHOOSE ... in RHS of $\langle 5\rangle 8$.

$\langle 5\rangle 10.$ Q.E.D.

$\langle 6\rangle 1.$ $r \triangleq \text{CHOOSE } i \in \mathit{Nat} : \wedge\ \mathit{IsIndex}(i,\tau,p,v)$
$\qquad\qquad\qquad\qquad\qquad\qquad \wedge\ \neg \mathit{IsIndex}(i+1,\tau,p,v)$

$\langle 6\rangle 2.$ $\wedge\ r \in \mathit{Nat}$
$\wedge\ \mathit{IsIndex}(r,\tau,p,v)$
$\wedge\ \neg \mathit{IsIndex}(r+1,\tau,p,v)$
BY $\langle 5\rangle 7$, ZF axiom about CHOOSE .

$\langle 6\rangle 3.$ $\wedge\ \tau[k][\![\langle x,h,m\rangle]\!]_{J|_{f\triangleq p}} = \tau[r][\![\langle x,h,m\rangle]\!]_{J|_{f\triangleq p}}$
$\wedge\ \tau[k][\![\langle x,h,m\rangle]\!]_{J|_{f\triangleq p}} \neq \tau[r+1][\![\langle x,h,m\rangle]\!]_{J|_{f\triangleq p}}$
BY $\langle 6\rangle 2$, $\langle 5\rangle 3$, $\langle 1\rangle 12$.

$\langle 6\rangle 4.$ $\tau[r][\![\langle x,h,m\rangle]\!]_{J|_{f\triangleq p}} \neq \tau[r+1][\![\langle x,h,m\rangle]\!]_{J|_{f\triangleq p}}$
BY $\langle 6\rangle 3$.

$\langle 6\rangle 5.$ $\tau[k] =_{x,h,m} \tau[r]$
BY $\langle 6\rangle 3$, x, h, m VARIABLES , $\langle 6\rangle 2$ ($r \in \mathit{Nat}$), $\langle 3\rangle 2$/A ($k \in \mathit{Nat}$), $\langle 1\rangle 21$, $\langle 1\rangle 9$ ($\mathit{IsABehavior}(\tau)$), DEF $=_{a,b,\ldots}$.

$\langle 6\rangle 6.$ $\tau[r] \neq_{x,h,m} \tau[r+1]$
BY $\langle 6\rangle 4$, x, h, m VARIABLES , $\langle 3\rangle 2$/A ($k \in \mathit{Nat}$), $\langle 1\rangle 21$, $\langle 1\rangle 9$ ($\mathit{IsABehavior}(\tau)$), DEF $=_{a,b,\ldots}$.

$\langle 6\rangle 7.$ $\tau[k] \neq_{x,h,m} \tau[k+1]$
BY $\langle 5\rangle 2$, x, h, m VARIABLES , $\langle 3\rangle 2$/A ($k \in \mathit{Nat}$), $\langle 1\rangle 21$, $\langle 1\rangle 9$ ($\mathit{IsABehavior}(\tau)$), DEF $=_{a,b,\ldots}$.

$\langle 6\rangle 8.$ $\wedge\ \tau[r] =_{x,h,m} \tau[k]$
$\wedge\ \tau[r] \neq_{x,h,m} \tau[r+1]$
$\wedge\ \tau[k] \neq_{x,h,m} \tau[k+1]$
BY $\langle 6\rangle 5$, $\langle 6\rangle 6$, $\langle 6\rangle 7$.

$\langle 6\rangle 9.$ $r = k$
BY $\langle 6\rangle 8$, $\langle 1\rangle 24$, $\langle 6\rangle 1$ ($r \in \mathit{Nat}$), $\langle 3\rangle 2$/A ($k \in \mathit{Nat}$), $\langle 1\rangle 21$, $\langle 1\rangle 9$ ($\mathit{IsABehavior}(\tau)$).

$\langle 6\rangle 10.$ Q.E.D.

BY $\langle 6\rangle 9$, $\langle 6\rangle 1$, $J|_{f \triangleq p}, \tau \models \wedge\ m = \langle f, 0\rangle$ $\qquad$ ($\langle 3\rangle 1$/A,
$\qquad\qquad \wedge\ \Box[m' = \langle f, m[2]+1\rangle]_{\langle m,x,h\rangle}$
$\langle 1\rangle 23$, $\langle 1\rangle 22$, $\langle 1\rangle 6/1$—similarly to $\langle 1\rangle 25/\langle 2\rangle 1/\langle 3\rangle 4$) $\wedge$E, $\wedge$I.

$\langle 4\rangle 5$. Q.E.D.
BY $\langle 4\rangle 4$.

$\langle 2\rangle 3$. Q.E.D.
BY $\langle 2\rangle 1$, $\langle 2\rangle 2$.

$\langle 1\rangle 26$. $R|_{f \triangleq p}, \sigma \models \exists\, x, h : \vee\ \neg \mathit{IsAFunction}(f)$
$\qquad \vee\ \wedge\ \exists\, m : \Box[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}$
$\qquad\quad \wedge\ \neg\varphi(x, h)$

$\langle 2\rangle 1$. $R|_{f \triangleq p}, \sigma \models \exists\, x, h, m : \vee\ \neg \mathit{IsAFunction}(f)$
$\qquad \vee\ \wedge\ \Box[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}$
$\qquad\quad \wedge\ \neg\varphi(x, h)$

$\langle 3\rangle 1$. $R|_{f \triangleq p}, \tau \models \vee\ \neg \mathit{IsAFunction}(f)$
$\qquad \vee\ \wedge\ \Box[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}$
$\qquad\quad \wedge\ \neg\varphi(x, h)$
BY $\langle 1\rangle 25$.

$\langle 3\rangle 2$. $\wedge\ \mathit{IsABehavior}(\tau)$
$\wedge\ \sigma \sim_{x,h,m} \tau$
BY $\langle 1\rangle 21$, $\langle 1\rangle 9$, $\wedge$E.

$\langle 3\rangle 3$. $\wedge\ \mathit{IsABehavior}(\tau)$
$\wedge\ \sigma \sim_{x,h,m} \tau$
$\wedge\ R|_{f \triangleq p}, \tau \models \vee\ \neg \mathit{IsAFunction}(f)$
$\qquad \vee\ \wedge\ \Box[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}$
$\qquad\quad \wedge\ \neg\varphi(x, h)$
BY $\langle 3\rangle 1$, $\langle 3\rangle 2$, $\wedge$I.

$\langle 3\rangle 4$. Q.E.D.
BY $\langle 3\rangle 3$, DEF $\exists$.

$\langle 2\rangle 2$. Q.E.D.

$\langle 3\rangle 1$. $R|_{f \triangleq p}, \sigma \models \exists\, x, h : \vee\ \neg \mathit{IsAFunction}(f)$
$\qquad \vee\ \exists\, m : \wedge\ \Box[x' = f[\langle x, h, m\rangle]]_{\langle x,h,m\rangle}$
$\qquad\qquad \wedge\ \neg\varphi(x, h)$

$\langle 4\rangle 1$. ASSUME: 1. TEMPORAL A, B
2. m does not occur in A
PROVE: $\models (\exists\, m : A \vee B) \equiv (A \vee \exists\, m : B)$

$\langle 4\rangle 2$. m does not occur in $\neg \mathit{IsAFunction}(f)$
BY DEFS $\mathit{IsAFunction}$, f (constant), m (variable).

$\langle 4\rangle 3$. Q.E.D.
BY $\langle 2\rangle 1$, $\langle 4\rangle 1$, $\langle 4\rangle 2$.

$\langle 3\rangle 2$. Q.E.D.

$\langle 4\rangle 1$. ASSUME: 1. TEMPORAL A, B

2. m does not occur in A
PROVE: $\models (\exists\, m : A \wedge B) \equiv (A \wedge \exists\, m : B)$

$\langle 4\rangle 2$. m does not occur in $\varphi(x, h)$

BY ASSUMPTION 6.

$\langle 4 \rangle 3.$ Q.E.D.

BY $\langle 3 \rangle 1$, $\langle 4 \rangle 1$, $\langle 4 \rangle 2$.

$\langle 1 \rangle 27.$ Q.E.D.

BY $\langle 1 \rangle 26$, ASSUMPTION 8 (DEF ψ), $\langle 1 \rangle 19$.

[Proposition 7 on page 26] PROOF:

$\langle 1 \rangle 1.$ PICK $\sigma : \land\ IsABehavior(\sigma)$
$\qquad\qquad \land\ \sigma \models \land\ x = \text{TRUE}$
$\qquad\qquad\qquad\qquad \land\ y \notin \text{BOOLEAN}$
$\qquad\qquad\qquad\qquad \land\ \psi$

BY DEF ψ, Proposition 2 (simplified realizations exist from arbitrary initial conditions).

$\langle 1 \rangle 2.$ $PrefixSat(\sigma, 0, A) \equiv \text{TRUE}$

$\langle 2 \rangle 1.$ $PrefixSat(\sigma, 0, A) \equiv \exists_{\text{behavior}}\, \tau : \tau \models \land\ y = \text{FALSE}$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ \Box(x = \text{TRUE})$

PROOF:

$$\begin{aligned}
PrefixSat(0, A) &= \exists_{\text{behavior}}\, \tau : \land\ \forall\, i \in 0..(-1) : \tau[i] = \sigma[i] \\
&\qquad\qquad\qquad\ \land\ \tau \models A \\
&\equiv \exists_{\text{behavior}}\, \tau : \land\ \forall\, i \in \emptyset : \tau[i] = \sigma[i] \\
&\qquad\qquad\qquad\ \land\ \tau \models A \\
&\equiv \exists_{\text{behavior}}\, \tau : \tau \models \land\ y = \text{FALSE} \\
&\qquad\qquad\qquad\qquad\ \land\ \Box(x = \text{TRUE})
\end{aligned}$$

$\langle 2 \rangle 2.$ $\exists_{\text{behavior}}\, \tau : \tau \models \land\ y = \text{FALSE}$
$\qquad\qquad\qquad\qquad \land\ \Box(x = \text{TRUE})$

BY WITNESS $\tau \triangleq [n \in Nat \mapsto [var \in VarNames \mapsto (var \neq \text{“}y\text{”})]]$

$\langle 2 \rangle 3.$ Q.E.D.

BY $\langle 2 \rangle 1$, $\langle 2 \rangle 2$.

$\langle 1 \rangle 3.$ $PrefixSat(\sigma, 1, G) \equiv \text{FALSE}$

$\langle 2 \rangle 1.$ $PrefixSat(\sigma, 1, G) \Rightarrow \exists_{\text{behavior}}\, \tau : \tau \models \land\ x = \text{TRUE}$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ y \notin \text{BOOLEAN}$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ \Box[\land\ y \in \text{BOOLEAN}\ ]_{\langle x, y \rangle}$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ y' = x$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ \Box\Diamond(y = x)$

$$\begin{aligned}
\mathit{PrefixSat}(\sigma, 1, G) &= \exists_{\text{behavior}}\, \tau \,:\, \begin{array}{l} \wedge\ \forall\, i \in 0..(1-1) \,:\, \tau[i] = \sigma[i] \\ \wedge\ \tau \models G \end{array} \\
&\equiv \exists_{\text{behavior}}\, \tau \,:\, \begin{array}{l} \wedge\ \forall\, i \in 0..0 \,:\, \tau[i] = \sigma[i] \\ \wedge\ \tau \models G \end{array} \\
&\equiv \exists_{\text{behavior}}\, \tau \,:\, \begin{array}{l} \wedge\ \tau[0] = \sigma[0] \\ \wedge\ \tau \models G \end{array} \\
&\Rightarrow \exists_{\text{behavior}}\, \tau \,:\, \tau \models \begin{array}{l} \wedge\ x = \textsc{true} \\ \wedge\ y \notin \textsc{boolean} \\ \wedge\ \Box[\begin{array}{l}\wedge\ y \in \textsc{boolean} \\ \wedge\ y' = x\end{array}]_{\langle x,y \rangle} \\ \wedge\ \Box\Diamond(y = x) \end{array}
\end{aligned}$$

$\langle 2 \rangle 2.$ SUFFICES: ASSUME: $\exists_{\text{behavior}}\, \tau \,:\, \tau \models \begin{array}{l} \wedge\ x = \textsc{true} \\ \wedge\ y \notin \textsc{boolean} \\ \wedge\ \Box[\begin{array}{l}\wedge\ y \in \textsc{boolean} \\ \wedge\ y' = x\end{array}]_{\langle x,y \rangle} \\ \wedge\ \Box\Diamond(y = x) \end{array}$

PROVE: FALSE

BY $\langle 2 \rangle 1$.

$\langle 2 \rangle 3.$ PICK $\tau \,:\, \tau \models \begin{array}{l} \wedge\ x = \textsc{true} \\ \wedge\ y \notin \textsc{boolean} \\ \wedge\ \Box[\begin{array}{l}\wedge\ y \in \textsc{boolean} \\ \wedge\ y' = x\end{array}]_{\langle x,y \rangle} \\ \wedge\ \Box\Diamond(y = x) \end{array}$

BY $\langle 2 \rangle 2$.

$\langle 2 \rangle 4.$ $\Box(x \neq y)$

$\langle 3 \rangle 1.$ $\tau \models \Box(y \notin \textsc{boolean}\ )$

BY $\langle 2 \rangle 3$, INV1.

$\langle 3 \rangle 2.$ $\tau \models \Box(x = \textsc{true})$

$\langle 4 \rangle 1.$ $\tau \models \begin{array}{l} \wedge\ x = \textsc{true} \\ \wedge\ \Box[\textsc{false}]_{\langle x,y \rangle} \end{array}$

BY $\langle 2 \rangle 3$, $\langle 3 \rangle 1$.

$\langle 4 \rangle 2.$ Q.E.D.

BY $\langle 4 \rangle 1$, INV1.

$\langle 3 \rangle 3.$ Q.E.D.

BY $\langle 3 \rangle 1$, $\langle 3 \rangle 2$.

$\langle 2 \rangle 5.$ $\tau \models \begin{array}{l} \wedge\ \Box(x \neq y) \\ \wedge\ \Box\Diamond(x = y) \end{array}$

BY $\langle 2 \rangle 3$, $\langle 2 \rangle 4$.

$\langle 2 \rangle 6.$ Q.E.D.

BY $\langle 2 \rangle 5$.

$\langle 1 \rangle 4.$ $\sigma \models \neg A \overset{+}{\rhd} G$

$\langle 2\rangle 1.$ $\sigma \models A \overset{+}{-\!\!\triangleright} G \equiv \land\ \sigma \models (A \Rightarrow G)$
$\qquad\qquad\qquad\qquad \land\ \mathit{PrefixSat}(\sigma, 0, A) \Rightarrow \mathit{PrefixSat}(\sigma, 1, G)$
$\qquad\qquad\qquad\qquad \land\ \forall\, n \in \mathit{Nat} : \mathit{PrefixSat}(\sigma, n, A) \Rightarrow \mathit{PrefixSat}(\sigma, n+1, G)$
BY DEF $\overset{+}{-\!\!\triangleright}$ (page 34), $\forall$E with $n = 0$.
$\langle 2\rangle 2.$ $(\mathit{PrefixSat}(\sigma, 0, A) \Rightarrow \mathit{PrefixSat}(\sigma, 1, G)) \equiv$ FALSE
BY $\langle 1\rangle 2$, $\langle 1\rangle 3$.
$\langle 2\rangle 3.$ Q.E.D.
BY $\langle 2\rangle 1$, $\langle 2\rangle 2$.
$\langle 1\rangle 5.$ Q.E.D.
$\langle 2\rangle 1.$ $\land\ \mathit{IsABehavior}(\sigma)$
$\qquad \land\ \sigma \models \psi$
BY $\langle 1\rangle 1$, $\land$E.
$\langle 2\rangle 2.$ $\land\ \mathit{IsABehavior}(\sigma)$
$\qquad \land\ \sigma \models \psi \land \neg\varphi$
BY $\langle 2\rangle 1$, $\langle 1\rangle 4$, DEF φ, $\land$I.
$\langle 2\rangle 3.$ Q.E.D.
BY $\langle 2\rangle 2$, $\exists$G, DEF $\exists_{\text{behavior}}$.

B Auxiliary results

Proposition 9 (Vacuity)
ASSUME: $\neg A$
PROVE: $A \overset{+}{-\!\!\triangleright} G$

$\langle 1\rangle 1.$ SUFFICES: ASSUME: 1. NEW σ
$\qquad\qquad\qquad\qquad\qquad$ 2. $\mathit{IsABehavior}(\sigma)$
$\qquad\qquad\qquad$ PROVE: $\land\ \sigma \models A \Rightarrow G$
$\qquad\qquad\qquad\qquad\quad \land\ \forall\, n \in \mathit{Nat} : \mathit{PrefixSat}(\sigma, n, A) \Rightarrow \mathit{PrefixSat}(\sigma, n+1, G)$
BY DEF $\overset{+}{-\!\!\triangleright}$.
$\langle 1\rangle 2.$ $\forall\, n \in \mathit{Nat} : \mathit{PrefixSat}(\sigma, n, A) \equiv$ FALSE
BY DEF $\mathit{PrefixSat}$, ASSUMPTION .
$\langle 1\rangle 3.$ $\forall\, n \in \mathit{Nat} : \mathit{PrefixSat}(\sigma, n, A) \Rightarrow \mathit{PrefixSat}(\sigma, n+1, G)$
BY $\langle 1\rangle 2$.
$\langle 1\rangle 4.$ $\sigma \models A \Rightarrow G$
BY ASSUMPTION , $\lor$I.
$\langle 1\rangle 5.$ Q.E.D.
BY $\langle 1\rangle 3$, $\langle 1\rangle 4$, $\land$I.

Proposition 10 (Closed system specification)
ASSUME: A
PROVE: $G \equiv A \overset{+}{-\!\!\triangleright} G$

$\langle 1\rangle 1.$ SUFFICES: ASSUME: 1. NEW σ
$\qquad\qquad\qquad\qquad\qquad$ 2. $\mathit{IsABehavior}(\sigma)$

$$\textsc{Prove:}\quad \sigma \models G \equiv \begin{array}{l} \wedge\ \sigma \models A \Rightarrow G \\ \wedge\ \forall n \in Nat : \begin{array}{l} \vee\ \neg PrefixSat(\sigma, n, A) \\ \vee\ PrefixSat(\sigma, n+1, G) \end{array} \end{array}$$

BY DEF $\stackrel{+}{\rhd}$.

⟨1⟩2. $\forall n \in Nat : PrefixSat(\sigma, n, A) \equiv$ TRUE
BY DEF *PrefixSat*, ASSUMPTION .

⟨1⟩3.

$$(\forall n \in Nat : (PrefixSat(\sigma, n, A) \Rightarrow PrefixSat(\sigma, n+1, G))) \equiv$$
$$(\forall n \in Nat : PrefixSat(\sigma, n+1, G))$$

BY ⟨1⟩2.

⟨1⟩4. $(\sigma \models A \Rightarrow G) \ \equiv\ (\sigma \models G)$
BY ASSUMPTION .

⟨1⟩5. $(\sigma \models G) \Rightarrow (\sigma \models \forall n \in Nat : PrefixSat(\sigma, n+1, G))$
BY DEF *PrefixSat*, WITNESS $\tau \triangleq \sigma$.

⟨1⟩6. $\sigma \models G \equiv \begin{array}{l} \wedge\ \sigma \models A \Rightarrow G \\ \wedge\ \sigma \models \forall n \in Nat : PrefixSat(\sigma, n+1, G) \end{array}$
BY ⟨1⟩4, ⟨1⟩5.

⟨1⟩7. Q.E.D.
BY ⟨1⟩3, ⟨1⟩6.

Let[24]

$$SatFromAnyInitG \ \triangleq\ \boldsymbol{\forall} u : Init_G(u) \Rightarrow \boldsymbol{\exists} r : (r = u) \wedge G(r)$$

The predicate *SatFromAnyInitG* means that G is satisfiable from every initial state that satisfies $Init_G$.

Proposition 11 (**$PrefixSat(\sigma, 1, G)$**)
ASSUME:

1. VARIABLE x
2. STATE $Init_G(x)$
3. TEMPORAL $G(x)$
4. *no variables other than x occur in $Init_G(x), G(x)$*
5. ZF NEW σ
6. $IsABehavior(\sigma)$

$$\textsc{Prove:}\quad \left(\begin{array}{l} \wedge\ \sigma \models Init_G(x) \\ \wedge\ SatFromAnyInitG \end{array}\right) \Rightarrow PrefixSat(\sigma, 1, G(x))$$

⟨1⟩1. $PrefixSat(\sigma, 1, G(x)) = \exists_{\text{behavior}}\, \tau : \begin{array}{l} \wedge\ \tau[0] = \sigma[0] \\ \wedge\ \tau \models G(x) \end{array}$

[24]Instead of temporal quantification ($\boldsymbol{\forall} u$), we could have used rigid quantification ($\forall u$). Showing that the two alternatives are equivalent (using [30, Note 16, p.920] and the assumptions of Proposition 11) would only complicate the proofs, without any significant benefit.

```
  BY DEF $PrefixSat$.
⟨1⟩2. SUFFICES: ASSUME: $\land\ \sigma \models Init_G(x)$
                        $\land\ SatFromAnyInitG$
                PROVE:  $\exists_{\text{behavior}}\,\tau : \land\ \tau[0] = \sigma[0]$
                                              $\land\ \tau \models G(x)$
  BY DP, ⟨1⟩1.
⟨1⟩3. $\lor\ \neg\sigma \models Init_G(x)$
      $\lor\ \sigma \models \boldsymbol{\exists}\, r : (r = x) \land G(r)$
  ⟨2⟩1. $\models \boldsymbol{\forall}\, u : \lor\ \neg Init_G(u)$
                     $\lor\ \boldsymbol{\exists}\, r : (r = u) \land G(r)$
    BY ⟨1⟩2, $\land$E, DEF $SatFromAnyInitG$.
  ⟨2⟩2. $\forall_{\text{behavior}}\,\eta : \eta \models \lor\ \neg Init_G(x)$
                              $\lor\ \boldsymbol{\exists}\, r : (r = x) \land G(r)$
    BY ⟨2⟩1, DEF $\boldsymbol{\exists}$, ASSUMPTION 1.
  ⟨2⟩3. Q.E.D.
    BY ⟨2⟩2, ASSUMPTION 6, $\forall$E.
⟨1⟩4. $\sigma \models \boldsymbol{\exists}\, r : (r = x) \land G(r)$
  BY ⟨1⟩2, $\land$E, ⟨1⟩3, MP.
⟨1⟩5. $\exists_{\text{behavior}}\,\tau : \land\ \sigma \sim_r \tau$
                    $\land\ \tau \models (r = x) \land G(r)$
  BY ⟨1⟩4, DEF $\boldsymbol{\exists}$.
⟨1⟩6. PICK $\eta : \land\ IsABehavior(\eta)$
             $\land\ \sigma \sim_r \eta$
             $\land\ \eta[0][\![r = x]\!]$
             $\land\ \eta \models G(r)$
  BY ⟨1⟩5.
⟨1⟩7. $\land\ IsABehavior(\eta)$
      $\land\ \forall\, v \in VarNames \setminus \{\text{“}r\text{”}\} : \sigma[0][v] = \eta[0][v]$
      $\land\ \eta[0][\![r]\!] = \eta[0][\![x]\!]$
      $\land\ \eta \models G(r)$
  BY ⟨1⟩6, DEF $\sim_r$, $\forall$E.
⟨1⟩8. $\land\ IsABehavior(\eta)$
      $\land\ \eta[0][\![r]\!] = \sigma[0][\![x]\!]$
      $\land\ \eta \models G(r)$
  BY ⟨1⟩7, $\eta[0][\![r]\!] = \eta[0][\![x]\!] = \sigma[0][\![x]\!]$.
⟨1⟩9. $\tau \ \triangleq\ [n \in Nat \mapsto$ IF $n = 0$ THEN $\sigma[0]$
                                  ELSE $[\eta[n]$ EXCEPT $![\![x]\!] = \eta[n][\![r]\!]]\ ]$
⟨1⟩10. $\land\ IsABehavior(\tau)$
       $\land\ \tau[0] = \sigma[0]$
       $\land\ \tau \models G(x)$
  ⟨2⟩1. $\tau[0] = \sigma[0]$
    BY ⟨1⟩9, ASSUMPTION 6.
  ⟨2⟩2. $IsABehavior(\tau)$
    BY ⟨1⟩9, ⟨1⟩6.
  ⟨2⟩3. $\tau \models G(x)$
```

```
        ⟨3⟩1. $\forall n \in \mathit{Nat} : \tau[n][\![x]\!] = \eta[n][\![r]\!]$
          ⟨4⟩1. $\tau[0][\![x]\!] = \eta[0][\![r]\!]$
             BY ⟨2⟩1, ⟨1⟩8.
          ⟨4⟩2. $\forall n \in \mathit{Nat} \setminus \{0\} : \tau[n][\![x]\!] = \eta[n][\![r]\!]$
             BY ⟨1⟩9.
          ⟨4⟩3. Q.E.D.
           BY ⟨4⟩1, ⟨4⟩2.
        ⟨3⟩2. $\wedge$ $\mathit{IsABehavior}(\eta)$
              $\wedge$ $\eta \models G(r)$
           BY ⟨1⟩8.
        ⟨3⟩3. Q.E.D.
           BY ⟨3⟩1, ⟨3⟩2, ⟨2⟩2, ASSUMPTION 4.
      ⟨2⟩4. Q.E.D.
         BY ⟨2⟩1, ⟨2⟩2, ⟨2⟩3, $\wedge$I.
    ⟨1⟩11. Q.E.D.
      BY ⟨1⟩10, $\exists$G with WITNESS $\tau \triangleq \tau$, DEF $\exists_{\text{behavior}}$.
```

It is worth noting that in Proposition 11 we needed the assumption *SatFromAnyInitG*, in order to prove the reverse from the direction to be proved in Proposition 12. In other words, it holds that $\mathit{PrefixSat}(\sigma, 1, G) \Rightarrow (\sigma \models \mathit{Init}_G(x))$, but

$$(\sigma \models \mathit{Init}_G(x)) \not\Rightarrow \mathit{PrefixSat}(\sigma, 1, G),$$

instead

$$\begin{array}{ll}(\wedge\ \sigma \models \mathit{Init}_G(x) & \Rightarrow \mathit{PrefixSat}(\sigma, 1, G).\\ \ \wedge\ \mathit{SatFromAnyInitG}) & \end{array}$$

Proposition 12 ($\overset{+}{\rhd}$ implies component initial condition)

ASSUME: *1.* TEMPORAL A, G

2. STATE Init_G

3. $\exists_{\text{behavior}}\, \tau : \tau \models A$

4. $\models G \Rightarrow \mathit{Init}_G$

PROVE: $(A \overset{+}{\rhd} G) \;\Rightarrow\; \mathit{Init}_G$

```
    ⟨1⟩1. SUFFICES: ASSUME: 1. ZF NEW $\sigma$
                            2. $\mathit{IsABehavior}(\sigma)$
                            3. $\sigma \models A \overset{+}{\rhd} G$
                    PROVE:  $\sigma \models \mathit{Init}_G$
      BY DP.
    ⟨1⟩2. $\mathit{PrefixSat}(\sigma, 0, A) \Rightarrow \mathit{PrefixSat}(\sigma, 1, G)$
      ⟨2⟩1. $\wedge$ $\sigma \models A \Rightarrow G$
            $\wedge$ $\forall n \in \mathit{Nat} : \mathit{PrefixSat}(\sigma, n, A) \Rightarrow \mathit{PrefixSat}(\sigma, n+1, G)$
        BY ⟨1⟩1/A, DEF $\overset{+}{\rhd}$.
      ⟨2⟩2. $\forall n \in \mathit{Nat} : \mathit{PrefixSat}(\sigma, n, A) \Rightarrow \mathit{PrefixSat}(\sigma, n+1, G)$
        BY ⟨2⟩1, $\wedge$E.
      ⟨2⟩3. Q.E.D.
```

BY ⟨2⟩2, ∀E with $n \triangleq 0 \in Nat$.

⟨1⟩3. $\exists_{\text{behavior}}\, \tau : \land\ \tau[0] = \sigma[0]$
$\qquad\qquad\qquad\qquad \land\ \tau \models G$

⟨2⟩1. $\mathit{PrefixSat}(\sigma, 0, A)$

⟨3⟩1. $\mathit{PrefixSat}(\sigma, 0, A) = \exists_{\text{behavior}}\, \tau : \land\ \forall\, i \in 0..(0-1) : \tau[i] = \sigma[i]$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ \tau \models A$

BY DEF *PrefixSat*.

⟨3⟩2. $\mathit{PrefixSat}(\sigma, 0, A) \equiv \exists_{\text{behavior}}\, \tau : \tau \models A$

BY ⟨3⟩1.

⟨3⟩3. Q.E.D.

BY ⟨3⟩2, ASSUMPTION 3.

⟨2⟩2. $\mathit{PrefixSat}(\sigma, 1, G)$

BY ⟨2⟩1, ⟨1⟩2, MP.

⟨2⟩3. $\mathit{PrefixSat}(\sigma, 1, G) = \exists_{\text{behavior}}\, \tau : \land\ \forall\, i \in 0..(1-1) : \tau[i] = \sigma[i]$
$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \land\ \tau \models G$

BY DEF *PrefixSat*.

⟨2⟩4. Q.E.D.

⟨2⟩2, ⟨2⟩3.

⟨1⟩4. $(\exists_{\text{behavior}}\, \tau : \tau[0] = \sigma[0] \land \tau \models G) \ \Rightarrow\ \sigma \models \mathit{Init}_G$

⟨2⟩1. SUFFICES: ASSUME: $\exists_{\text{behavior}}\, \tau : \land\ \tau[0] = \sigma[0]$
$\qquad\qquad\qquad\qquad\qquad\qquad \land\ \tau \models G$

PROVE: $\sigma \models \mathit{Init}_G$

BY DP.

⟨2⟩2. PICK $\tau : \land\ \mathit{IsABehavior}(\tau)$
$\qquad\qquad \land\ \tau[0] = \sigma[0]$
$\qquad\qquad \land\ \tau \models G$

BY ⟨2⟩1.

⟨2⟩3. $\land\ \tau[0] = \sigma[0]$
$\qquad \land\ \tau \models \mathit{Init}_G$

BY ⟨2⟩2, ASSUMPTION 4.

⟨2⟩4. $\land\ \tau[0] = \sigma[0]$
$\qquad \land\ \tau[0][\![\mathit{Init}_G]\!]$

BY ⟨2⟩3, ASSUMPTION 2.

⟨2⟩5. $\sigma[0][\![\mathit{Init}_G]\!]$

BY ⟨2⟩4.

⟨2⟩6. Q.E.D.

BY ⟨2⟩5, ⟨1⟩1, ASSUMPTION 2.

⟨1⟩5. Q.E.D.

BY ⟨1⟩3, ⟨1⟩4.

Proposition 13 (Violation of environment initial condition)

ASSUME: *1.* VARIABLE x

2. TEMPORAL $A, G(x)$

3. STATE $\mathit{Init}_A, \mathit{Init}_G(x)$

4. no variables other than x occur in $\mathit{Init}_G(x), G(x)$

```
          $5.\ \models A \Rightarrow Init_A$
PROVE:  $(\land\ Init_G(x)$              $\Rightarrow\ A \overset{+}{-\!\!\triangleright} G(x)$
         $\land\ \neg Init_A$
         $\land\ SatFromAnyInitG)$

⟨1⟩1. SUFFICES: ASSUME: 1. $SatFromAnyInitG$
                        2. ZF NEW $\sigma$
                        3. $IsABehavior(\sigma)$
                        4. $\sigma \models Init_G(x) \land \neg Init_A$
                PROVE:  $\sigma \models A \overset{+}{-\!\!\triangleright} G(x)$
  BY DP, TLA$^+$ semantics.
⟨1⟩2. SUFFICES: $\land\ \sigma \models A \Rightarrow G(x)$
                $\land\ \forall n \in Nat : PrefixSat(\sigma, n, A) \Rightarrow PrefixSat(\sigma, n+1, G(x))$
  BY DEF $\overset{+}{-\!\!\triangleright}$.
⟨1⟩3. $\sigma \models A \Rightarrow G(x)$
  ⟨2⟩1. $\sigma \models \neg Init_A$
    BY ⟨1⟩1.
  ⟨2⟩2. $\sigma \models A \Rightarrow Init_A$
    BY ASSUMPTION 5, DEF $\models$, ⟨1⟩1 $(IsABehavior(\sigma))$.
  ⟨2⟩3. $\sigma \models (\neg Init_A) \Rightarrow (\neg A)$
    BY ⟨2⟩3, TLA$^+$ semantics, contrapositive.
  ⟨2⟩4. $\neg \sigma \models A$
    BY ⟨2⟩1, ⟨2⟩3, MP.
  ⟨2⟩5. Q.E.D.
    BY ⟨2⟩4, $\lor$I.
⟨1⟩4. $\forall n \in Nat : PrefixSat(\sigma, n, A) \Rightarrow PrefixSat(\sigma, n+1, G(x))$
  ⟨2⟩1. $\sigma \models PrefixSat(\sigma, 0, A) \Rightarrow PrefixSat(\sigma, 1, G(x))$
    ⟨3⟩1. $\sigma \models Init_G(x) \land SatFromAnyInitG$
      BY ⟨1⟩1.
    ⟨3⟩2. $\sigma \models PrefixSat(\sigma, 1, G(x))$
      BY Proposition 11.
    ⟨3⟩3. Q.E.D.
      BY ⟨3⟩2, $\lor$I.
  ⟨2⟩2. $\forall n \in Nat \setminus \{0\} : PrefixSat(\sigma, n, A) \Rightarrow PrefixSat(\sigma, n+1, G(x))$
    ⟨3⟩1. SUFFICES: ASSUME: NEW $n \in Nat \setminus \{0\}$
                    PROVE:  $\neg PrefixSat(\sigma, n, A)$
      BY $\lor$I, $\forall$G.
    ⟨3⟩2. SUFFICES: ASSUME: $PrefixSat(\sigma, n, A)$
                    PROVE:  FALSE
    ⟨3⟩3. $\exists_{\text{behavior}}\, \tau : \land\ \forall i \in 0..(n-1) : \tau[i] = \sigma[i]$
                        $\land\ \tau \models A$
      BY ⟨3⟩2, DEF $PrefixSat$.
    ⟨3⟩4. PICK $\tau : \land\ IsABehavior(\tau)$
                   $\land\ \tau[0] = \sigma[0]$
                   $\land\ \tau \models A$
      BY ⟨3⟩1, ⟨3⟩3.
```

$\langle 3 \rangle 5.$ $\neg\tau \models A$

$\langle 4 \rangle 1.$ $\neg\sigma \models \mathit{Init}_A$

BY $\langle 1 \rangle 1$.

$\langle 4 \rangle 2.$ $\neg\tau \models \mathit{Init}_A$

$\langle 5 \rangle 1.$ $\tau[0] = \sigma[0]$

BY $\langle 3 \rangle 4$, $\wedge$E.

$\langle 5 \rangle 2.$ Q.E.D.

BY $\langle 5 \rangle 1$, $\langle 4 \rangle 1$, ASSUMPTION 3.

$\langle 4 \rangle 3.$ $\tau \models (\neg \mathit{Init}_A) \Rightarrow (\neg A)$

$\langle 5 \rangle 1.$ $\mathit{IsABehavior}(\tau)$

BY $\langle 3 \rangle 4$.

$\langle 5 \rangle 2.$ Q.E.D.

BY ASSUMPTION 5, DEF $\models$, $\langle 5 \rangle 1$, contrapositive.

$\langle 4 \rangle 4.$ Q.E.D.

BY $\langle 4 \rangle 2$, $\langle 4 \rangle 3$, MP.

$\langle 3 \rangle 6.$ Q.E.D.

BY $\langle 3 \rangle 4$, $\langle 3 \rangle 5$.

$\langle 2 \rangle 3.$ Q.E.D.

BY $\langle 2 \rangle 1$, $\langle 2 \rangle 2$.

$\langle 1 \rangle 5.$ Q.E.D.

BY $\langle 1 \rangle 3$, $\langle 1 \rangle 4$, $\wedge$I.

Proposition 14

ASSUME: *1. RealizationWrong defined using Fig. 5 with* $w' = f[v]$ *and* $w = f[v]$.

2. OnlyAllowedChanges omitted from definition of realizability (similar proof can be carried out if included, using $f \triangleq \langle \text{CHOOSE } p \in A : p \neq x, 1 \rangle$ *in Proposition 14)*

3. ZF NEW σ

4. $\mathit{IsABehavior}(\sigma)$

5. $\sigma \models \wedge\ \mathit{RealizationWrong}(m)$
$\quad\wedge\ m = 0$
$\quad\wedge\ x \in A$
$\quad\wedge\ y \in B$

6. $w \triangleq \langle x, m \rangle$

7. $v \triangleq \langle m, x, y \rangle$

8. $\mu \triangleq \neg\text{UNCHANGED } \langle m, x, y \rangle$

PROVE: FALSE

$\langle 1 \rangle 1.$ $f \triangleq [r \in \{0\} \times A \times B \mapsto \langle 0 \rangle]$
$m_0 \triangleq 0$

$\langle 1 \rangle 2.$ $\sigma \models \wedge\ m = 0$
$\quad\wedge\ \Box[w' = f[v]]_v$
$\quad\wedge\ \vee\ \Box\Diamond\langle \text{TRUE} \rangle_v$
$\quad\quad\vee\ \Box\Diamond(w = f[v])$.

$\langle 2\rangle 1.$ $\models \mathit{RealizationWrong}(m) = \land\ m = 0$
$\qquad\qquad\qquad\qquad\qquad \land\ \Box[w' = f[v]]_v$
$\qquad\qquad\qquad\qquad\qquad \land\ \lor\ \Box\Diamond\langle \text{TRUE}\rangle_v$
$\qquad\qquad\qquad\qquad\qquad\quad \lor\ \Box\Diamond(w = f[v]).$

BY DEF

$$\begin{aligned} \mathit{RealizationWrong}(m) &\equiv \land\ m = m_0 \\ &\quad \land\ \Box[\neg\text{UNCHANGED } v \Rightarrow w' = f[v]]_v \\ &\quad \land\ \lor\ \Box\Diamond\langle \neg\text{UNCHANGED } v\rangle_v \\ &\quad\quad \lor\ \Box\Diamond w = f[v]. \\ &\equiv \land\ m = 0 \\ &\quad \land\ \Box[w' = f[v]]_v \\ &\quad \land\ \lor\ \Box\Diamond\langle \text{TRUE}\rangle_v \\ &\quad\quad \lor\ \Box\Diamond(w = f[v]). \end{aligned}$$

$\langle 2\rangle 2.$ $\sigma \models \mathit{RealizationWrong}(m)$

$\langle 2\rangle 3.$ Q.E.D.

BY $\langle 2\rangle 1$, $\langle 2\rangle 2$.

$\langle 1\rangle 3.$ $\sigma \models \Box(f[v] = \langle 0\rangle)$

$\langle 2\rangle 1.$ $\sigma \models \land\ \langle m, x, y\rangle \in \{0\} \times A \times B$
$\qquad\qquad \land\ \Box[\langle x, m\rangle' = f[\langle m, x, y\rangle]]_v$

BY $\langle 1\rangle 2$, ASSUMPTION 5, ASSUMPTION 6.

$\langle 2\rangle 2.$ $\models (\land\ \langle m, x, y\rangle \in \{0\} \times A \times B\ \Rightarrow\ (\langle m, x, y\rangle \in \{0\} \times A \times B)'$
$\qquad\quad \land\ [\langle x, m\rangle' = f[\langle m, x, y\rangle]]_v)$

$\langle 3\rangle 1.$ SUFFICES: ASSUME: $\land\ \langle m, x, y\rangle \in \{0\} \times A \times B$
$\qquad\qquad\qquad\qquad\qquad \land\ [\langle x, m\rangle' = f[\langle m, x, y\rangle]]_v)$
$\qquad\qquad\qquad\quad$ PROVE: $(\langle m, x, y\rangle \in \{0\} \times A \times B)'$

BY DP.

$\langle 3\rangle 2.$ $f[\langle m, x, y\rangle] = \langle 0\rangle$

$\langle 4\rangle 1.$ $\langle m, x, y\rangle \in \{0\} \times A \times B$

BY $\langle 3\rangle 1$/A.

$\langle 4\rangle 2.$ DOMAIN $f = \{0\} \times A \times B$

BY $\langle 1\rangle 1$.

$\langle 4\rangle 3.$ $\langle m, x, y\rangle \in$ DOMAIN f

BY $\langle 4\rangle 1$, $\langle 4\rangle 2$.

$\langle 4\rangle 4.$ Q.E.D.

BY $\langle 4\rangle 3$, $\langle 1\rangle 1$.

$\langle 3\rangle 3.$ $\langle x, m\rangle' \neq f[\langle m, x, y\rangle]$

$\langle 4\rangle 1.$ $\langle x, m\rangle' \neq \langle 0\rangle$

BY AXIOM about function equality.

$\langle 4\rangle 2.$ Q.E.D.

BY $\langle 4\rangle 1$, $\langle 3\rangle 2$.

$\langle 3\rangle 4.$ $v' = v$

BY $\langle 3\rangle 1$/A, $\langle 3\rangle 3$.

$\langle 3\rangle 5.$ Q.E.D.

BY $\langle 3\rangle 1$/A, $\langle 3\rangle 4$.

$\langle 2\rangle 3.$ $\models (\land\ \langle m,x,y\rangle \in \{0\}\times A\times B \quad \Rightarrow\ \Box(\langle m,x,y\rangle \in \{0\}\times A\times B)$
$\quad\land\ \Box[\langle x,m\rangle' = f[\langle m,x,y\rangle]]_v)$
BY $\langle 2\rangle 2$, INV1.
$\langle 2\rangle 4.$ $\sigma \models \Box(\langle m,x,y\rangle \in \{0\}\times A\times B)$
BY $\langle 2\rangle 1$, $\langle 2\rangle 3$, ASSUMPTION 4.
$\langle 2\rangle 5.$ Q.E.D.
BY $\langle 2\rangle 4$, $\langle 1\rangle 1$.
$\langle 1\rangle 4.$ $\sigma \models \land\ \Box[\text{FALSE}]_v$
$\quad\land\ \Box\Diamond\langle\text{TRUE}\rangle_v$
$\langle 2\rangle 1.$ $\sigma \models \land\ \Box(f[v] = \langle 0\rangle)$
$\quad\land\ \Box[w' = f[v]]_v$
$\quad\land\ \Box\Diamond \lor\ \langle\text{TRUE}\rangle_v$
$\quad\quad\lor\ w = f[v]$
BY $\langle 1\rangle 2$, $\langle 1\rangle 3$.
$\langle 2\rangle 2.$ $\sigma \models \land\ \Box[\text{FALSE}]_v$
$\quad\land\ \Box\Diamond \lor\ \langle\text{TRUE}\rangle_v$
$\quad\quad\lor\ \text{FALSE}$
BY $\langle 2\rangle 1$, ASSUMPTION 6, AXIOM about function equality.
$\langle 2\rangle 3.$ Q.E.D.
BY $\langle 2\rangle 2$.
$\langle 1\rangle 5.$ Q.E.D.
BY $\langle 1\rangle 4$.

Proposition 15 (f should control variables, not a tuple)

ASSUME: *1.* $w \triangleq \langle x,m\rangle$
2. $\mu \triangleq \neg\text{UNCHANGED}\ \langle m,x,y\rangle$
3. $(\langle x,y\rangle \notin A\times B) \Rightarrow \varphi$
4. RealizationWrong defined using Fig. 5 with $w' = f[v]$ *and* $w = f[v]$.
5. OnlyAllowedChanges omitted from definition of realizability (similar proof can be carried out if included, using $f \triangleq \langle \text{CHOOSE}\ p \in A : p \neq x, 1\rangle$ *in Proposition 14)*
6. IsRealizableWrong defined with similar change as RealizationWrong

PROVE: *IsRealizableWrong*

$\langle 1\rangle 1.$ SUFFICES: ASSUME: 1. NEW σ
2. $\mathit{IsABehavior}(\sigma)$
PROVE: $\sigma \models \mathit{RealizationWrong}(m) \Rightarrow \varphi$
BY DEF *IsRealizableWrong*.
$\langle 1\rangle 2.$ CASE $\sigma \models \langle x,y\rangle \notin A\times B$
BY ASSUMPTION 3.
$\langle 1\rangle 3.$ CASE $\sigma \models \langle x,y\rangle \in A\times B$
BY Proposition 14.
$\langle 1\rangle 4.$ Q.E.D.
BY $\langle 1\rangle 2$, $\langle 1\rangle 3$, which are exhaustive.

Unmentioned variables are irrelevant Assume that the formula φ mentions only finitely many variables. A behavior assigns values to *all* variables [8, p.18, p.313]. So, some variable z does not occur in φ. Assume σ is a behavior and $\sigma \models \varphi$. For any set S, let

$$Tau(S) \triangleq [\sigma \text{ EXCEPT } ![0] = [\sigma[0] \text{ EXCEPT } ![\![z]\!] = S]].$$

The operator *Tau* is injective. This proves that the collection of behaviors σ that satisfy $\sigma \models \varphi$ is too large to be a set [8, p.66, p.69].

Let *PhiVars* be the set of variables that occur in the formula φ. A variable not in *PhiVars* can be ignored for deciding whether $\sigma \models \varphi$. In other words, in order to design a winning strategy f, it suffices to reason only about variables in *PhiVars*. Define the projection operator

$$(17) \qquad P(state) \triangleq \text{CHOOSE } s : \begin{array}{l} \land\ IsAFunction(s) \\ \land\ \text{DOMAIN } s = PhiVars \\ \land\ \forall\, var \in PhiVars : s[var] = state[var] \end{array}$$

The operator P maps each state *state* to an assignment of values to variables in *PhiVars* that agrees with *state*. This operator can be used to show that semantic reasoning about sequences of assignments to *PhiVars* suffices for finding functions that implement TLA$^+$ properties.

C Lamport's definition of realizability

For quick reference, the definition given in [1] is repeated below, almost verbatim. Read [1] first, because the accompanying description has been omitted here.

$$\begin{array}{rcl}
H & \triangleq & \land\ h = \langle v \rangle \\
 & & \land\ \Box[h' = h \circ \langle v' \rangle]_{\langle h,v \rangle} \\
 & & \land\ \forall\, n : (n \in Nat \Rightarrow \Diamond(|h| > n)) \\
Strategy(\mu,\ f) & \triangleq & \boldsymbol{\forall}\, h : \left(\begin{array}{l} \land\ h \in (VAL^n) \\ \land\ |h| \neq 0) \end{array} \Rightarrow \begin{array}{l} (\lor\ \mu(last(h)/v, f(h)/v') \\ \lor\ f(h) = \bot) \end{array}\right. \\
Outcome(\mu,\ f) & \triangleq & \boldsymbol{\exists}\, h : \begin{array}{l} \land\ H \\ \land\ \Box[\,\mu \Rightarrow v' = f(h)\,]_{\langle h,v \rangle} \\ \land\ \lor\ \Box\Diamond\langle \mu \rangle_{\langle h,v \rangle} \\ \quad\ \lor\ \Box\Diamond f(h) = \bot \end{array} \\
RealizablePart(\mu,\ \varphi) & \triangleq & \exists f : \begin{array}{l} \land\ \boldsymbol{\forall}\, v : Outcome(\mu, f) \Rightarrow \varphi \\ \land\ Outcome(\mu, f) \end{array}
\end{array}$$

References

[1] L. Lamport, "Miscellany," 21 April 1991, note sent to TLA mailing list. [Online]. Available: http://lamport.org/tla/notes/91-04-21.txt

[2] Y. Kesten, N. Piterman, and A. Pnueli, "Bridging the gap between fair simulation and trace inclusion," *Information and Computation*, vol. 200, no. 1, pp. 35–61, 2005.

[3] N. Piterman, A. Pnueli, and Y. Sa'ar, "Synthesis of reactive(1) designs," in *VMCAI*, 2006, pp. 364–380.

[4] R. Bloem, B. Jobstmann, N. Piterman, A. Pnueli, and Y. Sa'ar, "Synthesis of reacive(1) designs," *JCSS*, vol. 78, no. 3, pp. 911–938, 2012.

[5] L. Lamport and L. C. Paulson, "Should your specification language be typed?" *TOPLAS*, vol. 21, no. 3, pp. 502–526, May 1999.

[6] L. Lamport, "Types are not harmless," SRC, DEC, Tech. Rep., 18 July 1995. [Online]. Available: http://lamport.org/tla/types.ps.Z

[7] ——, "Types considered harmful," SRC, DEC, Tech. Rep., 23 December 1992. [Online]. Available: http://lamport.org/tla/notes/types.ps.Z

[8] ——, *Specifying Systems: The TLA$^+$ language and tools for hardware and software engineers.* Addison-Wesley, 2002.

[9] M. Abadi, L. Lamport, and P. Wolper, "Realizable and unrealizable specifications of reactive systems," in *ICALP*, 1989, pp. 1–17.

[10] M. Abadi and L. Lamport, "An old-fashioned recipe for real time," *TOPLAS*, vol. 16, no. 5, pp. 1543–1571, 1994.

[11] L. Lamport, "How to write a 21st century proof," *Journal of fixed point theory and applications*, vol. 11, no. 1, pp. 43–63, 2012.

[12] H.-D. Ebbinghaus, *Ernst Zermelo: An approach to his life and work.* Springer, 2007.

[13] K. Kunen, *Set theory: An introduction to independence proofs.* North Holland, 1980.

[14] H. Vanzetto, "Proof automation and type synthesis for set theory in the context of TLA$^+$," Ph.D. dissertation, Université de Lorraine, Dec 2014. [Online]. Available: https://hal.inria.fr/tel-01096518

[15] T. Nipkow, M. Wenzel, and L. C. Paulson, *Isabelle/HOL: A proof assistant for higher-order logic.* Springer, 2002.

[16] S. Merz, "File `isabelle/Functions.thy` in TLAPS v1.4.3," LORIA, 2011, time-stamp: 2011-10-11 17:38:56 merz. [Online]. Available: http://tla.msr-inria.inria.fr/tlaps/dist/current/tlaps-1.4.3.tar.gz

[17] ——, "Logic-based analysis of reactive systems: Hiding, composition, and abstraction," Habilitationsschrift, Ludwig-Maximilians-Universität, München, Dec 2001. [Online]. Available: http://www.loria.fr/~merz/papers/habil.ps.gz

[18] A. C. Leisenring, *Mathematical Logic and Hilbert's ε-symbol.* MacDonald Technical & Scientific, 1969.

[19] M. Abadi, "An axiomatization of Lamport's Temporal Logic of Actions," Digital Equipment Corporation, Tech. Rep., 1993. [Online]. Available: http://www.hpl.hp.com/techreports/Compaq-DEC/SRC-RR-65.pdf

[20] S. Merz, "On the completeness of Propositional Raw TLA," 1997. [Online]. Available: http://www4.in.tum.de/~merz/isabelle/TLA/doc/ptla.ps

[21] D. Hilbert and P. Bernays, *Grundlagen der Mathematik II.* Springer, 1970.

[22] T. Y. Chow, "A beginner's guide to forcing," in *Communicating mathematics*, ser. Contemporary Mathematics. AMS, 2009, vol. 479, pp. 25–40. [Online]. Available: https://arxiv.org/abs/0712.1320

[23] J. Bagaria, "Set theory," in *The Stanford Encyclopedia of Philosophy*, winter 2016 ed., E. N. Zalta, Ed. Metaphysics Research Lab, Stanford University, 2016.

[24] K. Kunen, *Set theory*, ser. Studies in Logic. College Publications, 2013, vol. 34.

[25] L. Lamport, "Errata to Specifying Systems," 28 Oct 2016. [Online]. Available: http://lamport.org/tla/errata.pdf

[26] ——, "TLA^{+2}: A preliminary guide," Tech. Rep., 15 Jan 2014. [Online]. Available: https://research.microsoft.com/en-us/um/people/lamport/tla/tla2-guide.pdf

[27] M. Abadi and L. Lamport, "Conjoining specifications," *TOPLAS*, vol. 17, no. 3, pp. 507–535, 1995.

[28] S. Merz, "A user's guide to TLA," in *Modélisation et vérification des processus parallèles: Actes de l'école d'été.* Nantes, France: Ecole centrale de Nantes, 1998, pp. 29–44.

[29] A. Pnueli and U. Klein, "Synthesis of programs from temporal property specifications," in *MEMOCODE*, 2009, pp. 1–7.

[30] L. Lamport, "The temporal logic of actions," *TOPLAS*, vol. 16, no. 3, pp. 872–923, 1994.

[31] S. Merz, *The specification language TLA$^+$.* Springer, 2008, pp. 401–451.