A Formalized Theory for Verifying Stability and Convergence of Automata in PVS*

Sayan Mitra and K. Mani Chandy

California Institute of Technology
Pasadena, CA 91125
{mitras,mani}@caltech.edu

Abstract. Correctness of many hybrid and distributed systems require stability and convergence guarantees. Unlike the standard induction principle for verifying invariance, a theory for verifying stability or convergence of automata is currently not available. In this paper, we formalize one such theory proposed by Tsitsiklis [26]. We build on the existing PVS metatheory for untimed, timed, and hybrid input/output automata, and incorporate the concepts about fairness, stability, Lyapunov-like functions, and convergence. The resulting theory provides two sets of sufficient conditions, which when instantiated and verified for particular automata, guarantee convergence and stability, respectively.

1 Introduction

Verification of many classes of systems require proofs for stability and convergence. For example, the requirement that a hybrid control system regains equilibrium in the face of disturbances is a stability property; the requirement that a set of mobile agents get arbitrarily close to the centroid of their initial positions through interaction is a convergence property. To best of our knowledge, existing frameworks that formalize automata in higher-order logics do not define these notions nor do they provide sufficient conditions for verifying them. In this paper we present a PVS [22] metatheory for stating and verifying stability and convergence properties. This theory extends the PVS interface for the Tempo toolkit [1]; thus, along with invariance properties and implementation relations, now we can also prove stability and convergence of automata, within the same framework.

In a 1987 paper [26] Tsitsiklis analyzed stability and convergence of a general class of models which he called Asynchronous Iterative Processes (henceforth, AIPs). An AIP consists of a set X, and a finite collection of functions or "operators" $T_k : X \to X$, $k \in \{1, \ldots, K\}$. Given an initial point $x_0 \in X$, an execution is obtained by choosing an arbitrary sequence of $T_k's$, and iteratively applying them to x_0. An AIP is *stable* around a given point $x^* \in X$, with respect to a given topological structure $\mathscr{T}$ on X, if for every neighborhood set $U \in \mathscr{T}$

* The work is funded in part by the Caltech Information Science and Technology Center and AFOSR MURI FA9550-06-1-0303.

containing x^*, there exists another neighborhood set $V \in \mathscr{T}$, such that every execution starting from V remains within U. An infinite execution is said to be *fair* if every operator T_k is applied infinitely many times. An AIP *converges* to x^* with respect to a topological structure $\mathscr{T}$ around x^*, if for every $U \in \mathscr{T}$ containing x^*, there exists $n \in \mathbb{N}$, such that for every fair execution all the states obtained after applying the first n operations, are in U. In general, neither of these properties imply the other (see, Figure 1 for examples). In [26] the author provides sufficient conditions for proving stability and convergence of AIPs in terms of Lyapunov-like functions [15]. Moreover, under some weak assumptions about the topological structures, it turns out that these conditions are also necessary.

AIPs generalized to infinite (and possibly uncountable) set of operations subsume the classes of discrete, timed, and hybrid automata, and therefore, the sufficient conditions for proving convergence and stability of AIPs apply to these classes as well. In this paper, we formalize Tsitsiklis' theory of stability and convergence in PVS. We build on the existing PVS metatheory for untimed, timed, and hybrid I/O automata [3,14] which is integrated with the Tempo toolkit [1]. The preexisting theory defines reachable states and implementation relations for automata and provides theorems for inductively verifying invariant properties and simulation relations. We extend this metatheory as follows:

(a) A real-valued distance function d on pairs of states is introduced as a parameter to the theory; for a given state s^*, the sublevel sets of the function $d(s^*, .)$ define a topological structure around s^*.
(b) Infinite executions of automata and fairness conditions are defined.
(c) Stability and convergence of automata are defined with respect to a given state (or a set of states), a distance function, and a fairness condition.
(d) A set of theorems are stated and proved which provide sufficient conditions for verifying convergence and stability.

The new metatheory can be downloaded from `http://ist.caltech.edu/~mitras/research/pvs/convergence/`. In order to apply the theory to particular automata, the user has to supply a Lyapunov-like function and check that this function satisfies the criteria prescribed in the theorems. This check can be performed by analyzing the state transitions of the automaton and one need not reason about infinite executions. We illustrate the application of the theory on a simple distributed coordination protocol.

A key issue in formalizing this metatheory was to reconcile fairness of AIPs with the notion of fair executions of automata. Recall that an AIP-execution is fair, if *every* operator is applied infinitely often. This definition is too strong for automata, where each operator corresponds to a *specific* state transition. A fair execution for an automaton typically does not have every state transition occurring infinitely many times, instead it has some set of transitions represented infinitely often. More precisely, fairness of an automaton is defined with respect to a collection $\mathcal{F} = \{F_1, F_2, \ldots, F_n\}$, where each F_i is a set of transitions. An execution α is said to be $\mathcal{F}$-fair if every $F_i \in \mathcal{F}$ is represented (or scheduled) in α infinitely often. In our theory, we formalize this weaker notion of fairness, and

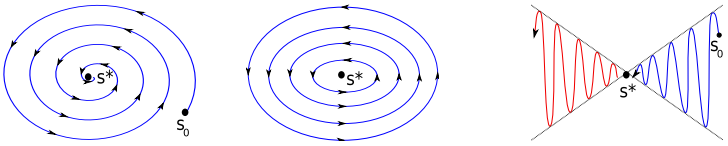

Fig. 1. Stable & convergent (*left*), stable & nonconvergent (*middle*), and convergent & unstable (*right*) executions. In the last case, the execution from s_0 converges, but executions starting from in the left neighborhood of s^* diverge.

hence, the sufficient conditions we obtain are stronger than those in [26]. If each F_i is defined to be a singleton transition then our conditions reduce to Tsitsiklis'. There are several other relatively minor differences between the original theory and our PVS formalization; these are discussed in Section 6.

2 Related work

Convergence of general sequences has been formalized in the PVS and Isabelle libraries for differential calculus [9], real-analysis [10], and topology [12]. There are several formalizations of automata in higher-order logics of theorem provers including Isabelle/HOL [20,21,19,24], PVS [3,6,18], and Coq [7,23]. These theories formalize reachable states, invariant properties, acceptance conditions, and abstraction relations, but neither stability nor convergence. Our theory builds on the PVS formalization of input/output automata [16] presented in [3] and its subsequent extensions to timed and hybrid I/O automata [11] that were presented in [18,17].

Literature on stability and convergence (also called asymptotic stability) of purely discrete or continuous systems is extensive. Control theory textbooks, such as [15], typically provide conditions for checking stability of processes evolving in Euclidean space. Stability conditions for hybrid and switched systems is an active area of research; we refer the reader to [13] for an overview.

Although convergence is distinct from termination, constructing proofs for both these properties rely on existence of Lyapunov-like functions from the state space of the automaton to some well-ordered set. There is a large body of literature on proving termination of programs and recursive functions using theorem provers (see, e.g., [4,25]). This research direction focuses on automatically finding Lyapunov-like functions that prove termination. This connection is further discussed in Section 6.

3 Automata and Executions

3.1 Preliminaries

In this paper, we present our PVS metatheory using standard set theoretic notations, in as much of a syntax-free manner as possible. The set theoretic definitions, in most cases, correspond in an obvious way to the definitions in PVS's

simply typed higher-order logic. Wherever necessary we note special constructs that are necessary for this translation.

We denote the set of boolean constants by $\mathbb{B} = \{true, false\}$, the set of natural numbers by $\mathbb{N} = \{0, 1, \ldots, \}$, and the set of reals by $\mathbb{R}$. For a set A, A^ω is defined as the set of infinite sequences of elements in A indexed by $\mathbb{N}$. For $a^\omega \in A^\omega, i \in \mathbb{N}$, we denote the i^{th} element by a^ω_i. In the rest of this section we summarize the relevant definitions and theorems from the existing metatheory of [3].

3.2 Formalization of Automata

An automaton $\mathscr{A}$ is a nondeterministic, labeled transition system. Formally, it is a quintuple consisting of:

(a) a nonempty set S,
(b) a nonempty set A,
(c) a nonempty subset S_0 of S,
(d) a function $E : [S, A \to \mathbb{B}]$, and
(e) a function $T : [S, A \to S]$.

Elements of S, S_0, and A are called *states, starting states* and *actions*, respectively. E and T are called the *enabling predicate* and the *transition function* of $\mathscr{A}$. The actions are labels for state transitions. For $s \in S$ and $a \in A$, $E(s, a)$ holds if and only if the transition labeled by a can be applied to s. In this case, a is said to be *enabled* at s. At any state s, multiple actions may enabled. However, once an action a is fixed the post-state of the transition s' is uniquely determined. Specifically, $s' = T(a, s)$, if a is enabled at s, or else $s' = s$.

The set of actions can be uncountably infinite and indeed actions can label functions for discrete transitions as well as continuous evolution. We refer the reader to [14] for models of timed and hybrid systems in this formalism.

The PVS metatheory formalizing automata is parameterized by $\mathcal{S}$, $\mathcal{A}$, $\mathcal{S}_0$, $\mathcal{E}$ and $\mathcal{T}$, where $\mathcal{S}$ and $\mathcal{A}$ are uninterpreted type parameters, and $\mathcal{S}_0, \mathcal{E}$, and $\mathcal{T}$ are parameters with the appropriate type constraints[1]. To apply the metatheory to specific systems, the parameters are instantiated with concrete types and function definitions. The following example shows such an instantiation.

Example 1. We model a distributed algorithm in which a set of N agents start at arbitrary positions on a line and through interactions converge to a point. Specifically, any two agents interact at any time; when they do, their positions are atomically updated so that they move towards each other by some fraction of the distance between them.

Table 1 provides concrete definitions for the types and the functions for modeling this protocol as an automaton. N is a constant natural number. L is a constant real in the range $(0, 1)$. I is the type $\{0, \ldots, N-1\}$. The states type is an array of $\mathbb{R}_{\geq 0}$ indexed by I. For any state s and $i \in I$, the i^{th} component of

[1] Each parameter corresponds to its unscripted version in our presentation of the automaton theory. E.g., the set of states S is modeled as the type $\mathcal{S}$ in PVS.

the array is denoted by $s[i]$. The state s_0 is an arbitrary but constant element of S. The predicate S_0 defines a state s to be a starting state if and only if it equals s_0. The action type is defined using a (datatype) constructor called *interact*. This construct means that for every $i, j \in I$, $r \in [L, 1-L]$ $interact(i, j, r)$ is an action; and nothing else is an action. The set of actions is uncountably infinite because of the real parameter r.

The enabling predicate $E(s, a)$ returns true for any action a and state s, which means that agents i and j can interact always. Finally, for action a of the form $interact(i, j, r)$, the state transition function $T(a, s)$ returns a state s' that is identical to s except that the i^{th} and the j^{th} values of s' are $s[i]+r.(s[i]-s[j])$ and $s[j]-r.(s[i]-s[j])$, respectively. Informally, for given i, j, each choice of r defines a different proportion by which agents i and j move towards each other. For example, $interact(i, j, \frac{1}{2})$ causes $s[i]$ and $s[j]$ to move to their mid-point.

S : Type := array$[I \mapsto \mathbb{R}_{\geq 0}]$
s_0 : Const S
$S_0(s : S)$: *bool* := $(s = s_0)$

A : Datatype $\quad interact(i, j : I, r : [L, 1-L])$

$E(s : S, a : A)$: *bool* := true

$T(s : S, a : A) : S :=$ [Case $a \equiv interact(i, j, r)$:
$\quad s$ With $[(i) := s[i] + r(s[j] - s[i]), (j) := s[j] - r(s[j] - s[i])]]$

Table 1. An instance of automaton metatheory

3.3 Executions, Reachability, and Invariance

The semantics of an automaton $\mathscr{A}$ is defined in terms of its executions. An *execution fragment* of $\mathscr{A}$ is a (possibly infinite) alternating sequence of states and actions $s_0, a_0, s_1, a_1, s_2, \ldots$, such that for each i, $E(s_i, a_i)$ holds and $s_{i+1} = T(a_i, s_i)$. An execution fragment is an *execution* if s_0 is a starting state. The *length* of a finite execution is the number of actions in it. For $s \in S$ and a natural number n, $Reach\_rec(s, n)$ returns true if and only if there exists an execution of length n that ends in the state s. The reachability predicate on states is defined recursively as follows:

$$Reach\_rec(s, n) := \begin{cases} s \in S_0 & n = 0 \\ \exists\ s_1 \in S, a \in A\ (E(a, s_1) \wedge & \text{otherwise.} \\ \quad s = T(a, s_1)\ \wedge\ Reach\_rec(s_1, n-1)) & \end{cases}$$

$$Reach(s) := \exists\ n \in \mathbb{N},\ Reach\_rec(s, n).$$

An *invariant* of $\mathscr{A}$ is a predicate on its states that holds in all reachable states. Invariants are useful for capturing safety requirements, such as, multiple

processes *never* access a critical resource simultaneously. The following theorem formalizes Floyd's induction principle [8] in this framework.

Theorem 1. *Suppose* $G : [S \to \mathbb{B}]$ *is a predicate on* S*, and*

A1. $\forall\ s \in S_0$*,* $G(s)$*, and*
A2. $\forall\ s \in S, a \in A$*,* $Reach(s)\ \wedge\ E(a,s)\ \wedge\ G(s) \Rightarrow G(T(a,s))$*.*

Then $\forall\ s \in S$*,* $Reach(s) \Rightarrow G(s)$*, that is,* G *is an invariant predicate.*

This theorem has been employed for verifying safety properties of untimed [3], timed [14], and hybrid automata [27]. Features of this verification method that make it attractive are: (a) It suffices to check that the predicate G is preserved over individual actions, and hence, the check breaks down into a case analysis of actions. (b) This structure facilitates partial automation of proofs using customized proof strategies [2].

4 Formalizing Stability and Convergence of Automata

In this section we present the extensions to the PVS metatheory for stability and convergence verification. In order to define stability and convergence properties, first, we have to explicitly define arbitrary prefixes of infinite executions of the automaton. Given a state $s \in S$, an infinite sequence of actions $a^\omega \in A^\omega$, and $n \in \mathbb{N}$, the recursively defined $Trans(s, a^\omega, n)$ function returns the state that is obtained by applying the first n actions in a^ω to s.

$$Trans(s,a^\omega,n) = \begin{cases} s & \text{if } n = 0 \\ T(a^\omega_{n-1}, Trans(s,a^\omega,n-1)) & \text{if } E(a^\omega_{n-1}, Trans(s,a^\omega,n-1)) \\ Trans(s,a^\omega,n-1) & \text{otherwise} \end{cases}$$

Note that s, a^ω, and n, uniquely determine an execution fragment $s_0, a^\omega_0, s_1, \ldots, a^\omega_{n-1}, s_n$ of length n, where $s_i = Trans(s, a^\omega, i)$, for each $i < n$.

Stability and convergence of automaton $\mathscr{A}$ to a state $s^* \in S$ are defined with respect to a topological structure around s^*. This topological structure is formalized using a real-valued function. The metatheory can be easily generalized to define stability (and convergence) with respect to arbitrarily defined topological structures around a point; this is discussed further in Section 6.

Definition 1. *A* distance function d *for a state* $s^* \in S$ *is a real-valued function* $d : [S, S \to \mathbb{R}_{\geq 0}]$*, such that for all* $s \neq s^*$*,* $d(s^*, s) > d(s^*, s^*)$*. A* distance function d *for a set of states* $S^* \subseteq S$ *is a real-valued function* $d : [2^S, S \to \mathbb{R}_{\geq 0}]$*, such that for all* $s \notin S^*, s' \in S^*$*,* $d(S^*, s) > d(S^*, s')$*.*

For $\epsilon > 0$ *and* $s \in S$*,* ϵ-ball around s*, is the set*

$$B_\epsilon(s) := \{s_1 \in S \mid d(s_1, s) \leq \epsilon\}.$$

In our theory, d is not required to satisfy identity, symmetry, and triangle inequality, properties that are usually attributed to metrics. The ϵ-balls around a given state s define a topological structure around s. In the new PVS metatheory we add S^* and d as theory parameters, in addition to the six parameters enumerated in Section 3.2.

4.1 Stability

Informally, automaton $\mathscr{A}$ is stable if every execution fragment that starts close to the equilibrium state s^* remains close to s^*, where closeness is defined in terms of the ϵ-balls of some distance function for s^*.

Definition 2. *Let $\mathscr{A}$ be an automaton $\langle S, A, S_0, E, T\rangle$, s^* be a state in S, and d be a distance function for s^*. $\mathscr{A}$ is* (s^*, d)-stable *if*

$$\forall\, \epsilon > 0, \exists\, \delta > 0,\ \forall s \in S, a^\omega \in A^\omega, n \in \mathbb{N}, s \in B_\delta(s^*) \Rightarrow Trans(s, a^\omega, n) \in B_\epsilon(s^*).$$

Note that stability is independent of the starting states of the automaton. For a nonempty set $S^* \subseteq S$, let d be a distance function for S^*. The definitions for the ϵ-balls around S^* (denoted by $B_\epsilon(S^*)$) and (S^*, d)-stability of $\mathscr{A}$ are analogous to Definition 2.

The coarseness of the topological structure around s^* (or S^*), and hence, the meaning of stability depends on the function d. For example, suppose $d(s^*, s) := 0$ if $s^* = s$, and $d(s^*, s) := 1$, otherwise. Then, $\mathscr{A}$ is trivially (s^*, d)-stable. On the other hand, if the set of states S is an Euclidean space and d is the Euclidean metric on S, then d defines an uncountable set of distinct ϵ-balls around s^*. And in this case (s^*, d)-stability of $\mathscr{A}$ depends on E and T.

4.2 Sufficient conditions for stability

In this section we present the sufficient conditions for proving stability of automaton $\mathscr{A}$. The proofs of all the theorems presented in Section 4 have been completed in PVS and are available as part of the metatheory. Here we present summaries of these PVS proofs.

Let T be a set and $<$ be a total order on T, and f be a function that maps S to T. The range of f is denoted by Rng_f, and the *p-sublevel set* is defined as $L_{f,p} := \{s : S \mid f(s) \leq p\}$. We omit the subscript f when the function is clear from the context. The following theorem gives a sufficient condition for proving stability of an automaton in terms of a Lypunov-like function.

Theorem 2. *Let S^* be a nonempty subset of S and d be a distance function for S^*. Suppose there exists a totally ordered set $(T, <)$ and a function $f : [S \to T]$ that satisfies the following conditions:*

B1. $\forall\, \epsilon \geq 0$, $\exists\, p \in T$, *such that* $L_p \subseteq B_\epsilon(S^*)$.
B2. $\forall\, p \in T$, $\exists\, \epsilon \geq 0$, *such that* $B_\epsilon(S^*) \subseteq L_p$.
B3. $\forall\, s \in S, a \in A,\ E(a, s) \Rightarrow f(T(a, s)) \leq f(s)$.

Then $\mathcal{A}$ is (S^*, d)-stable.

B1 requires that every ϵ-ball around S^* contains a p-sublevel set L_p. B2 is symmetric; it requires that every sublevel set contains an ϵ-ball. B3 states that the value of the function f does not increase if an action a is applied to state where it is enabled.

Proof: Let us fix an $\epsilon > 0$. We have to show that there exists a $\delta > 0$, such that any execution fragment that starts in $B_\delta(S^*)$ remains within $B_\epsilon(S^*)$. There exists $p \in T$, such that $L_p \subseteq B_\epsilon(S^*)$ (by B1), and there exists a $\eta \geq 0$, such that $B_\eta(S^*) \subseteq L_p \subseteq B_\epsilon(S^*)$ (by B2). Set $\delta = \eta$, and fix an $s \in B_\delta(S^*)$, $a^\omega \in A^\omega$. We show by induction that every state in the execution fragment starting from s and corresponding to a^ω remains within $B_\epsilon(S^*)$.

Base case: We know that $s \in B_\delta(S^*) \subseteq B_\epsilon(S^*)$.

Inductive step: Let s' be the state $Trans(s, a^\omega, j)$, $0 \leq j \leq n-2$. By the induction hypothesis, $s' \in B_\delta(S^*) \subseteq L_p$. If a^ω_j is not enabled at s', then $s_{j+1} = s' \in B_\delta(S^*) \subseteq B_\epsilon(S^*)$. Otherwise, $s_{j+1} = T(a_m, s')$. As $f(s_{j+1}) \leq f(s')$ (by B3), and it follows that $s_n \in L_p \subseteq B_\epsilon(S^*)$.

4.3 Fairness

An automaton $\mathscr{A}$ is said to *converge* to s^* with respect to distance function d, if for every infinite execution $s_0, a_0, s_1, \ldots, a_{n-1}, s_n \ldots$, $d(s^*, s_n) \to 0$ as $n \to \infty$. This captures the informal notion that every execution of the automaton gets closer and closer to s^*.

In typical applications, the above definition of convergence is too strong because it quantifies over all infinite executions—including those in which some set of actions never occur. For instance, consider an infinite execution α for the automaton of Example 1 that starts from a state s_0 with distinct values for all $s_0[i]$'s, and in which agent 0 never interacts with any other agent. It is easy to see that such an execution does not converge. On the other hand, a different infinite execution $\alpha' = s_0, a_0, s_1, \ldots$, converges to s^*, where $s^*[i] = \frac{1}{N}\sum_{i=1}^{N} s_0[i]$, provided for every $i, j \in I$, infinitely many $interact(i, j, *)$ actions occur in α'. In fact, an infinite execution in which for every $i \in I$, $interact(i, (i+1) \mod N, *)$ occurs infinitely often, also converges to s^*. This suggests that the convergence of $\mathscr{A}$ can be studied under different sets of assumptions about the occurrence of the actions. This motivates the following definition of fairness.

Definition 3. *A* fairness condition $\mathcal{F}$ *for the set of actions A is a finite collection $\{F_i\}_{i=1}^n$, $n \in \mathbb{N}$, where each F_i is a nonempty subset of A. An infinite sequence of actions $a^\omega \in A^\omega$ is $\mathcal{F}$-fair if*

$$\forall\ F \in \mathcal{F}, n \in \mathbb{N},\ \exists\ m \in \mathbb{N},\ m > n,\ \textit{such that } a^\omega_m \in F.$$

An infinite execution $\alpha = s_0, a_0, s_1, a_1, \ldots$ is said to $\mathcal{F}$-fair if the corresponding sequence of actions $a_0, a_1, \ldots$ is $\mathcal{F}$-fair.

In other words, an execution is not $\mathcal{F}$-fair if there exists $F \in \mathcal{F}$ such that no action from F ever appears in some suffix of α.

Definition 4. *Given fairness conditions $\mathcal{F}_1$ and $\mathcal{F}_2$ for the set of actions A, $\mathcal{F}_1$ is said to be* weaker *than $\mathcal{F}_2$, denoted by $\mathcal{F}_1 \leq \mathcal{F}_2$, if $\forall\ F_1 \in \mathcal{F}_1$, $\exists\ F_2 \in \mathcal{F}_2$, such that $F_2 \subseteq F_1$.*

The next lemma states that an $\mathcal{F}_2$-fair execution is also $\mathcal{F}_1$-fair, if $\mathcal{F}_1$ is a weaker fairness condition than $\mathcal{F}_2$.

Lemma 1. *Let $\mathcal{F}_1, \mathcal{F}_2$ be fairness conditions for the set of actions A. If $\mathcal{F}_1 \leq \mathcal{F}_2$, then every $\mathcal{F}_2$-fair execution is $\mathcal{F}_1$-fair.*

4.4 Convergence

Having introduced fairness of executions, we now modify the previously suggested definition of convergence as follows. Informally, automaton $\mathscr{A}$ *converges* to s^* with respect to distance function d and a fairness condition $\mathcal{F}$, if every $\mathcal{F}$-fair execution converges to s^*.

Definition 5. *Let $\mathscr{A}$ be an automaton $\langle S, A, S_0, E, T\rangle$, s^* be an element of S, d be a function for s^*, and $\mathcal{F}$ be a fairness condition for A. $\mathscr{A}$ is* $(s^*, d, \mathcal{F})$-convergent, *if $\forall\ s_0 \in S_0, \epsilon > 0, a^\omega \in A^\omega$*

$$\textit{if } a^\omega \textit{ is } \mathcal{F}\textit{-fair then } \exists n \in \mathbb{N}, \forall m \in \mathbb{N},\ m > n \Rightarrow Trans(s, a^\omega, m) \in B_\epsilon(s^*).$$

For a nonempty subset of states $S^* \subseteq S$, the definition of $(S^*, d, \mathcal{F})$-convergence is analogous to Definition 5. For $s \in S, a^\omega \in A^\omega$, we define $R_f(s, a^\omega)$ as the set of values in T that can be reached from s by applying some prefix of a^ω.

$$R_f(s, a^\omega) := \{p \in T \mid \exists\ n \in \mathbb{N}, Trans(s, a^\omega, n) \in L_{f,p}\}.$$

The next theorem gives sufficient conditions for proving convergence of automaton $\mathscr{A}$ in terms of a Lyapunov-like function.

Theorem 3. *Let S^* be a nonempty subset of S, d be a distance function for S^*, and $\mathcal{F}$ be a fairness condition on A. Suppose there exists a totally ordered set $(T, <)$ and a function $f : S \to T$ that satisfies the following conditions:*

C1. $\forall\ p, q \in T, p < q \Rightarrow L_p \subsetneq L_q$.
C2. $\forall\ \epsilon > 0$, $\exists\ p \in T$, *such that* $L_p \subseteq B_\epsilon(S^*)$.
C3. $\forall\ s \in S, a \in A$, $(Reach(s)\ \wedge\ E(a, s)) \Rightarrow f(T(a, s)) \leq f(s)$.
C4. $\forall\ p \in T$, $L_p \neq S^*$ *implies* $\exists\ F \in \mathcal{F}$, *such that* $\forall\ a \in F, s \in L_p$, $Reach(s) \Rightarrow (E(a, s)\ \wedge\ f(T(a, s)) < f(s))$.
C5. $\forall\ s \in S_0$ *and* $\mathcal{F}$*-fair sequence* $a^\omega \in A^\omega$, $R' \subseteq R_f(s, a^\omega)$, *if R' is lower bounded then it has a smallest element.*

Then $\mathscr{A}$ is $(S^*, d, \mathcal{F})$-convergent.

Some remarks about the hypothesis of the theorem are in order. C1 implies that every sublevel set of the function f is distinct. C2 requires that for any $\epsilon > 0$, there exists a p-sublevel set of f that is contained within the ϵ-ball around S^*. This is identical to condition B1. C3 requires that the function f is nonincreasing over all transitions from reachable states. This is a weaker version of B3. C4 requires that for any sublevel set L_p that is not equal to the convergence set S^*, there exists a fair set of actions $F \in \mathcal{F}$, such that any action $a \in F$ strictly decreases the value of f—possibly by some arbitrarily small amount. C5 requires that for all s, a^ω, every lower-bounded subset of $R_f(s, a^\omega)$ has a smallest element. This is a weaker assumption than requiring $R_f(s, a^\omega)$ to be well-ordered. Instead of C5 it is sometimes easier to prove that the set T is well-ordered.

Before proving Theorem 3, we state a set of intermediate lemmas that are used in the proof. In the following, S^* is a subset of S, $\mathcal{F}$ is a fairness condition on A, $(T, <)$ is a total order, and f is a function $S \to T$ satisfying conditions C1-5. We make the following assumption, without any loss of generality.

$$\exists\ p^* \in T,\ \text{such that}\ \forall\ s \in S,\ \text{if}\ s \in S^*\ \text{then}\ f(s) = p^*,\ \text{otherwise}\ f(s) > p^*.$$

This is without loss of generality because for any given f' we can define $f(x) := p^* = \inf_{s \in S^*} f(s)$ if $x \in S^*$ and $f'(x) := f(x)$ otherwise. Then f satisfies the assumption and we work with it instead of f'.

Lemma 2. *For all $a^\omega \in A^\omega, n \in \mathbb{N}, p \in T, s \in L_p$: $Trans(s, a^\omega, n) \in L_p$.*

Lemma 3. *For all $s \in S$, $s \in S^*$ iff $L_{f(s)} = S^*$.*

Lemma 4. *For all $s_0 \in S_0$ and $\mathcal{F}$-fair action sequence a^ω, $Rng_f = R_f(s_0, a^\omega)$.*

Proof: Let us fix an a^ω, s_0, and f. We abbreviate $R_f(s_0, a^\omega)$ as R. From its definition it is clear that $R \subseteq Rng_f$, so, we have to show that for every $p \in Rng$, there is an n such that $f(Trans(s_0, a^\omega, n)) \leq p$. Let us fix a value of $p \in Rng$, and suppose for the sake of contradiction that $p \notin R$. We know that $f(s_0) < p$, because otherwise $Trans(s_0, a^\omega, 0) = s_0 \in L_p$, that is, p would be in R. We consider two subcases:

Case 1: R has a lower bound. R has a smallest element, say p_{min} (by C5). If $p_{min} \leq p$ then $p \in R$, so, we consider the case where $p < p_{min}$. There exists p^* such that $f(s) = p^*$ for every $s \in S^*$, and $p^* < f(s)$ outside S^* (by Lemma 3).

Case 1.1: $p_{min} \leq p^*$. Then $p < p^*$ and this contradicts our assumption that p^* is the smallest value attained by f.

Case 1.2: $p^* < p < p_{min}$: Since $L_p \neq S^*$, there exists an $F \in \mathcal{F}$, such that for every $a \in F$ and for every reachable state s in L_p, a is enabled at s, and $f(T(a, s)) < p$ (by C4). Also, there exists n_0 such that $Trans(s_0, a^\omega, n_0)) \in L_{p_{min}}$ (by definition of p_{min}). Since a^ω is an $\mathcal{F}$-fair sequence, there exists an m, $m > n_0$, such that $a^\omega_m \in F$. We define $s' := Trans(s_0, a^\omega, m-1)$. It follows that $s' \in L_{p_{min}}$ (by Lemma 2), that is, $f(T(a^\omega_m, s')) < p_{min}$. As $f(T(a^\omega_m, s')) \in R$, this contradicts our assumption that p_{min} is the smallest element in R.

Case 2: R does not have a lower bound. Then there exists $q \in R$, such that $q < p$, and by C1, $L_q \subsetneq L_p$. That is, there exists n, such that $Trans(s_0, a^\omega, n) \in L_p$, and therefore, contrary to our assumption $p \in R$.

Proof of Theorem 3: Let us fix $\epsilon \geq 0$, f satisfying the conditions in the hypothesis, $s_0 \in S_0$, and an $\mathcal{F}$-fair sequence $a^\omega = a_0, a_1, \ldots$. There exists $p \in Rng_f$, such that $L_{f,p} \subseteq B_\epsilon(S^*)$ (by C2). There exists $n \in \mathbb{N}$, such that $Trans(s_0, a^\omega, n) \in L_{f,p} \subseteq B_\epsilon(S^*)$ (by Lemma 4). It follows that for all $m > n$, $Trans(s_0, a^\omega, m) \in L_{f,p} \subseteq B_\epsilon(S^*)$ (by Lemma 2).

Corollary 1. *If $\mathscr{A}$ is $(S^*, d, \mathcal{F}_1)$-convergent and $\mathcal{F}_1 \leq \mathcal{F}_2$ then $\mathscr{A}$ is $(S^*, d, \mathcal{F}_2)$-convergent.*

4.5 Special case

In certain applications the function d which defines the topological structure around s^* can itself be used as the Lyapunov-like function for proving convergence. The obvious advantage of doing so is that C2 follows automatically. We provide a restricted version of Theorem 3 which can be applied in such cases.

Corollary 2. *Let S^* be a nonempty subset of S and $\mathcal{F}$ be a fairness condition on A. We define $f : S \to \mathbb{R}_{\geq 0}$ as $f(s) := d(S^*, s)$. Suppose there exists a strictly decreasing sequence $p^\omega \in {\mathbb{R}_{\geq 0}}^\omega$ of valuations of f that converges to 0, such that:*

D1. $\forall\ i, j \in \mathbb{N}, i > j \Rightarrow L_{p_i} \subsetneq L_{p_j}$.
D2. $\forall\ s \in S, a \in A, i \in \mathbb{N}\ (Reach(s)\ \wedge\ E(a,s)\ \wedge\ s \in L_{p_i}) \Rightarrow T(a,s) \in L_{p_i}$.
D3. $\forall\ i \in \mathbb{N}$, $p_i \neq 0$ *implies* $\exists\ F \in \mathcal{F}$, *such that* $\forall\ a \in F, s \in L_{p_i}$, $Reach(s) \Rightarrow (E(a,s)\ \wedge\ T(a,s) \in L_{p_{i+1}})$.

Then $\mathscr{A}$ $(S^*, d, \mathcal{F})$-convergent.

Proof: : We check that the defined function f satisfies the conditions in the hypothesis of Theorem 3. C1 follows from D1. C2 follows from the convergence of the sequence p^ω. C3 follows from D2. C4 follows from D3 and the strictly decreasing property of the sequence p^ω. It remains to show C5.

Let us fix $s \in S_0$ and a $\mathcal{F}$-fair sequence $a^\omega \in A^\omega$, and let $p_{min} > 0$ be a lower bound for a subset $R' \subseteq R_f(s, a^\omega)$. Suppose, for the sake of contradiction, that R' does not have a smallest element. Then there exists a $i \in \mathbb{N}$ such that $p_i \in R'$, and for all $j > i$, $p_j < p_{min}$. There exists $s \in L_{p_i}$, such that $s' = Trans(s_0, a^\omega, m)$, for some $m \in \mathbb{N}$ (by definition of R'). There exists $F \in \mathcal{F}$, such that for all $a \in F, s \in L_{p_i}$, if s is reachable then $E(a,s)$ and $T(a,s) \in L_{p_{i+1}}$ (by D3). We fix such an F. As a^ω is an $\mathcal{F}$-fair sequence, there exists $k > m$, such that $a_k \in F$. Let $s' = Trans(s_0, a^\omega, k)$. As s' is reachable and so is $T(a_k, s')$. Since $f(T(a_k, s')) \leq p_{i+1} < p_i$, it contradicts our assumption.

5 An Application

In this section, we illustrate the application of our convergence theory to verify the convergence of the protocol introduced in Example 1. Recall, the set of states S is defined as arrays of $\mathbb{R}_{\geq 0}$ indexed by I, where $I = \{0, 1, \ldots, N-1\}$. For $i \in I$, $s[i]$ corresponds to the value of the i^{th} participating agent at state s. The starting state s_0 is an arbitrary but constant element of S. We define a real-valued constant M that corresponds to the average of the initial values of the agents: $M := \frac{1}{N} \sum_{j=0}^{N-1} s_0[j]$. Let $s^* \in S$ be defined as the constant array:

$$s^* : \text{Const } S := [M, \ldots, M]$$

We would like to prove convergence of this protocol to s^*, and in order to do so we have to first define some notion of neighborhood around s^* and a fairness condition for the actions of this automaton. We define the neighborhood around s^* with the standard Euclidean distance between any state $s \in S$ and s^*.

$$d(s^*, s) := \sum_j (s[j] - s^*[j])^2$$

Next, we specify the fairness condition $\mathcal{F}$. Informally, we require that no two sets of participating agents are partitioned perpetually. That is, an action sequence a^ω is $\mathcal{F}$-fair, if for every $n \in \mathbb{N}$, and for every pair of disjoint subsets $J, K \subset I$, there exists $m > n$, such that $a_m = interact(j, k, r)$ for some $j \in J, k \in K$ and $r \in [L, 1-L]$. Formally, let J, K be any two subsets of I. We define

$$F_{J,K} := \{interact(j, k, r) \mid j \in J,\ k \in K,\ r \in [L, 1-L]\ \},$$
$$\mathcal{F} := \{F_{J,K} \mid J \cap K = \emptyset\}.$$

Our goal is to prove that automaton $\mathscr{A}$ is $(s^*, d, \mathcal{F})$-convergent. To this end we invoke Corollary 2. In this case $S^* = \{s^*\}$. We define

$$f(s) := s(s^*, s) = \sum_j (s[j] - M)^2,$$

and the sequence p^ω, as $p_i = M\beta^i$, where $\beta := (1 - \frac{2L(1-L)}{N^3})$. It is easy to check that p^ω converges to 0 and that f satisfies condition D1. In the remainder of this section we check that f satisfies D2 and D3.

We define $sorted(s)$ as a derived variable that returns the sorted version of the array s. That is, $sorted(s) : [I \to \mathbb{R}_{\geq 0}]$ has the following ordering property. For all $i, j \in I$, $sorted(s)(i) < sorted(s)(j)$ if and only if $s[i] > s[j]$ or $s[i] = s[j]$ and $i > j$. That is, $sorted(s)[k]$ is the k^{th} largest element of s.

Prior to checking D2 and D3 the following invariant properties are verified using Theorem 1. The first invariant follows from the property of the protocol that $\sum_{i=0}^{N-1} s[i] = MN$ remains constant in all reachable states.

Invariant 4. *For every $s \in S$, $Reach(s)$ implies $\sum_{i=0}^{N-1} (sorted(s)[i] - M) = 0$.*

Invariant 5. *For every $s \in S$, $Reach(s)$ and $s \neq s^*$ implies*

$$\sum_{i=0}^{N-2} sorted(s)[i] - sorted(s)[i+1] \geq \sqrt{\frac{f(s)}{N}}.$$

Proof: Since $s \neq s^*$, $f(s) > 0$. There exists $j \in I$ such that $(s[j]-M)^2 \geq f(s)/N$ (by definition of $f(s)$). Let us fix such a j. Since $(s[j] - M)^2 \geq f(s)/N$ we conclude that $(sorted(s)[0]-M)^2 \geq f(s)/N$. We assume that $(sorted(s)[0]-M)^2 > 0$; proof for the negative case is symmetric. From Invariant 4, $\sum_{i=0}^{N-1}(sorted(s)[i]-M) = 0$ and the positivity of $sorted(s)[0] - M$ it follows that there exists $k \in I$, such that $sorted(s)[k] - M < 0$. Thus, $sorted(s)[N-1] - M < 0$. $\sum_{i=0}^{N-2}(sorted(s)[i] - sorted(s)[i+1])$

$$= sorted(s)[0] - sorted(s)[N-1]$$
$$= sorted(s)[0] - M - (sorted(s)[N-1] - M) \;\geq\; \sqrt{\frac{f(s)}{N}}.$$

Proposition 1. *f satisfies D2.*

Proof: We have to show that from every reachable state $s \in S$ and for any action $a \in A$, $f(T(a,s)) \leq f(s)$. Every action a of A, is of the form $interact(j,k,r)$, where $r \in [L, 1-L]$, and $L \in (0,1)$. We fix $j,k \in I$ and a reachable state $s \in S$, and define $\delta := s[k] - s[j]$. From the definition of the transition function T for action $interact(j,k,r)$ we know:

$$T(interact(j,k)s)[i] = \begin{cases} s[i] + \delta r & \text{if i = j} \\ s[i] - \delta r & \text{if i = k} \\ s[i] & \text{otherwise.} \end{cases} \tag{1}$$

Thus, $f(s) - f(T(interact(j,k,r),s)) = 2\delta^r(1-r)$, and since $r \in [L, 1-L]$, $f(s) - f(T(interact(j,k,r),s)) \geq 2.\delta^2.L.(1-L) \geq 0$

Proposition 2. *f satisfies D3.*

Proof: Consider any reachable state s such that $s \neq s^*$. It suffices to show that there exists $F \in \mathcal{F}$, such that for any action $a \in F$ $f(T(a,s)) \leq \beta f(s)$, where β has been defined to be $1 - \frac{2L(1-L)}{N^3}$. From Invariant 5 we know that $\sum_{i=0}^{N-2} sorted(s)[i] - sorted(s)[i+1] > \sqrt{\frac{f(s)}{N}}$. Since each term in the summation is nonnegative, we conclude that there exists $k \in I$, $sorted(s)[k] - sorted(s)[k+1] \geq \frac{1}{N}\sqrt{\frac{f(s)}{N}}$. We fix such a k and define two subsets of I:

$$A = \{j \in I \mid s[j] \geq s[k]\}$$
$$B = \{j \in I \mid s[j] \leq s[k-1]\}$$

Since A and B are disjoint subsets of I, $F_{A,B} \in \mathcal{F}$. Now we show that for any action $a \in F_{A,B}$, $f(T(a,s)) \leq \beta f(s)$. Let $a = interact(j,k,r)$, where $j \in A$ and $k \in B$. From Proposition 1 it follows that

$$\begin{aligned} f(s) - f(T(interact(j,k,r),s)) &\geq 2L(1-L)(s[j]-s[k])^2 \\ &\geq \frac{2f(s)L(1-L)}{N^3} \quad \text{(by definition of } A,B) \\ f(T(interact(j,k,r),s)) &\leq \left[1 - \frac{2L(1-L)}{N^3}\right] f(s). \end{aligned}$$

6 Discussions

Comparison with Tsitsiklis' theory. Apart from the more general notion of fairness that we have formalized, our theory for stability and convergence differs from that presented in [26] in the following ways.

Specifying topologies. In [26] closeness to the point of convergence s^* is defined in terms of a topological structure called a *neighborhood system* around s^*. A neighborhood system around s^* is a collection $\mathcal{U}$ of subsets of S that satisfies the following conditions: (i) $s^* \in U$, $\forall U \in \mathcal{S}$. (ii) For all $s \in S, s \neq s^*$, there exists $U \in \mathcal{U}$ such that $s \notin U$. (iii) $\mathcal{U}$ is closed under finite intersections and arbitrary unions. For most natural definitions for the distance function d, the ϵ-balls of d satisfy conditions $(i), (ii)$ and (iii). We decided to use this functional specification of the neighborhood sets because (a) it is concise, and (b) in many applications there exists an inherent metric with respect to which we prove convergence (or stability). Introducing neighborhood systems in the style of [26] would require us make relatively minor modifications to B1, B2, and C2.

Reachability. In [26] reachability conditions for states of AIPs are not introduced. Consequently, C4 and C5 in Theorem 3 are weaker than the corresponding conditions in [26]. This is because we require the f to be nonincreasing (decreasing, resp.) only from the reachable states. Thus, invariant properties proved using Theorem 1, can be used to verify these conditions.

Convergence and termination. The general method for proving termination is based on finding a function which decreases along every transition of an automaton. If the co-domain of the function is wellfounded, the automaton must terminate, because there are no infinite descending chains.

The standard definition of termination—that of an automaton or a program executing a finite sequence of transitions and then stopping with an answer—is not directly applicable to reactive system models where the automaton runs forever producing an infinite sequence of outputs. Thus, we redefine termination as follows: given a subset of states S^T, $\mathscr{A}$ *terminates* at S^T if for every execution $s_0, a_1, s_1, \ldots$, there exits $n \in \mathbb{N}$, such that for all $m > n$, $s_m \in S^T$. If we set

$S^* = S^T$, then this definition of termination is equivalent to the definition of convergence if one allows ϵ to be 0. With this interpretation, termination is a stronger property than convergence. Indeed, many distributed systems, such as the consensus protocol of Example 1, only convergence (for $\epsilon > 0$) can be guaranteed and not termination.

7 Conclusions

We have formalized fairness, stability, and convergence within an existing PVS framework for verifying untimed, timed, and hybrid automata. The theory provides a very general set of sufficient conditions for proving stability and convergence. These conditions can be checked using the PVS prover or using other tools. For example, the nonincreasing condition for convergence C3 can be checked with a model-checker. The theory extends the PVS interface for the Tempo toolkit, and hence, enables us to now verify invariance, implementation, convergence, and stability, all within the same software framework.

Currently we are applying the proposed metatheory to verify timed and hybrid system models; in particular, convergence of asynchronous pattern formation algorithms for mobile agent systems [5]. We plan on developing PVS strategies that exploit the common structural properties in these models and automate convergence and stability proofs.

References

1. Tempo toolset, version 0.2.2 beta, January 2008. http://www.veromodo.com/tempo/.
2. M. Archer. TAME: PVS Strategies for special purpose theorem proving. *Annals of Mathematics and Artificial Intelligence*, 29(1/4), February 2001.
3. M. Archer, C. Heitmeyer, and S. Sims. TAME: A PVS interface to simplify proofs for automata models. In *Proceedings of UITP '98*, July 1998.
4. L. Bulwahn, A. Krauss, and T. Nipkow. Finding lexicographic orders for termination proofs in isabelle/hol. In *TPHOLs*, volume 4732 of *Lecture Notes in Computer Science*, pages 38–53. Springer, 2007.
5. K. M. Chandy, S. Mitra, and C. Pilotto. Formations of mobile agents with message loss and delay, 2007. preprint http://www.ist.caltech.edu/~mitras/research/2008/asynchcoord.pdf.
6. M. Devillers. Translating IOA automata to PVS. Technical Report CSI-R9903, Computing Science Institute, University of Nijmegen, February 1999. Available at http://www.cs.ru.nl/research/reports/info/CSI-R9903.html.
7. J. Filliâtre. Finite automata theory in coq: A constructive proof of kleene's theorem. Technical report, LIP -ENS, Research Report 97-04, Lyon, February 1997.
8. R. Floyd. Assigning meanings to programs. In *Sympyposium on Applied Mathematics. Mathematical Aspects of Computer Science*, pages 19–32. American Mathematical Society, 1967.
9. H. Gottliebsen. Transcendental functions and continuity checking in pvs. In *TPHOLs '00: Proceedings of the 13th International Conference on Theorem Proving in Higher Order Logics*, pages 197–214, London, UK, 2000. Springer-Verlag.

10. J. Harrison. *Theorem Proving with the Real Numbers*. Springer-Verlag, 1998.
11. D. K. Kaynar, N. Lynch, R. Segala, and F. Vaandrager. *The Theory of Timed I/O Automata*. Synthesis Lectures on Computer Science. Morgan Claypool, November 2005. Also available as Technical Report MIT-LCS-TR-917.
12. D. Lester. NASA langley PVS library for topological spaces. http://shemesh.larc.nasa.gov/fm/ftp/larc/PVS-library/topology-details.html.
13. D. Liberzon. *Switching in Systems and Control*. Systems and Control: Foundations and Applications. Birkhauser, Boston, June 2003.
14. H. Lim, D. Kaynar, N. Lynch, and S. Mitra. Translating timed I/O automata specifications for theorem proving in pvs. In *Proceedings of Formal Modelling and Analysis of Timed Systems (FORMATS'05)*, number 3829 in LNCS, Uppsala, Sweden, September 2005. Springer.
15. D. G. Luenberger. *Introduction to Dynamic Systems: Theory, Models, and Applications*. John Wiley and Sons, Inc., New York, 1979.
16. N. Lynch and M. Tuttle. An introduction to Input/Output automata. *CWI-Quarterly*, 2(3):219–246, September 1989.
17. S. Mitra. *A Verification Framework for Hybrid Systems*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA 02139, September 2007.
18. S. Mitra and M. Archer. PVS strategies for proving abstraction properties of automata. *Electronic Notes in Theoretical Computer Science*, 125(2):45–65, 2005.
19. O. Müller. I/o automata and beyond: Temporal logic and abstraction in isabelle. In *Proceedings of the 11th International Conference on Theorem Proving in Higher Order Logics*, pages 331–348, London, UK, 1998. Springer-Verlag.
20. T. Nipkow and K. Slind. I/O automata in Isabelle/HOL. In P. Dybjer, B. Nordström, and J. Smith, editors, *Types for Proofs and Programs*, volume 996 of *LNCS*, pages 101–119. Springer, 1995.
21. Olaf Müller. *A Verification Environment for I/O Automata Based on Formalized Meta-Theory*. PhD thesis, Technische Universität München, Sept. 1998.
22. S. Owre, S. Rajan, J. Rushby, N. Shankar, and M. Srivas. PVS: Combining specification, proof checking, and model checking. In R. Alur and T. A. Henzinger, editors, *Computer-Aided Verification, CAV '96*, number 1102 in LNCS, pages 411–414, New Brunswick, NJ, July/August 1996. Springer-Verlag.
23. C. Paulin-Mohring. Modelisation of timed automata in coq. In *TACS '01: Proceedings of the 4th International Symposium on Theoretical Aspects of Computer Software*, pages 298–315, London, UK, 2001. Springer-Verlag.
24. L. C. Paulson. Mechanizing UNITY in Isabelle. *ACM Transactions on Computational Logic*, 1(1):3–32, 2000.
25. E. Rohwedder and F. Pfenning. Mode and termination checking for higher-order logic programs. In *ESOP '96: Proceedings of the 6th European Symposium on Programming Languages and Systems*, pages 296–310, London, UK, 1996. Springer-Verlag.
26. J. N. Tsitsiklis. On the stability of asynchronous iterative processes. *Theory of Computing Systems*, 20(1):137–153, December 1987.
27. S. Umeno and N. A. Lynch. Safety verification of an aircraft landing protocol: A refinement approach. In *HSCC*, pages 557–572, 2007.