ISSN 2007-9737

Analog Algorithms with Discontinuous States and Non-Unique Evolution Operators: Computability and Stability

Zvi Retchkiman Konigsberg

Instituto Politecnico Nacional,
Centro de Investigacion en Computacion,
Mexico

mzvi@cic.ipn.mx

Abstract. In this work computability and stability issues for analog algorithms with discontinuous states and non-unique evolution operators are studied. The notions of analog algorithm and dynamical system are postulated to be equivalent. The stability and stabilization concepts for analog algorithms are defined. The stability and stabilization presentation starts concentrating in continuous and discrete dynamical systems i.e., analog algorithms, defined by differential or difference equations, and continues considering Lyapunov energy functions for analog algorithms with continuous and discontinuous states. Dynamical systems with non-unique evolution operators are also studied.

Keywords. Analog algorithms, dynamical systems, discontinuous, non-unique evolution operators, stability, Lyapunov functions.

1 Introduction

This work presents analog algorithms with discontinuous states and non-unique evolution operators. Bournez, Dershowitz and Neron [1] have formalized a generic notion of analog algorithm. They provide postulates defining analog algorithms in the spirit of those given for discrete algorithms [2], and continue proving some completeness results. The notions of analog algorithm and dynamical system are postulated to be equivalent.

Retchkiman and Dershowitz [3] have studied the stability and stabilization concepts for analog algorithms. They first considered the stability and stabilization issues concentrating in continuous and discrete dynamical systems i.e., analog algorithms described by differential or difference equations. This paper extends these results considering Lyapunov energy functions for analog algorithms with continuous and discontinuous states which applies to many classes of dynamical systems including hybrid systems and switched systems. Dynamical systems with non-unique evolution operators are also presented.

The paper is organized as follows. In section 2, the paper written by Retchkiman and Dershowiz related to analog algorithms is first reviewed. The stability and stabilization concepts for analog algorithms are defined. Section 3 presents an application example. Section 4 discusses the stability concept for analog algorithms with continuous and discontinuous states in terms of Lyapunov energy functions, and finally section 5 discusses analog algorithms for dynamical systems with non-unique evolution operators.

2 Analog Algorithms, the Church-Turing, Stable Algorithms and Stabilization of Unstable Analog Algorithms

In this section, the work presented in [3] and the references therein is recalled.

Definition 1. A dynamical system is a four-tuple $\{T, X, A, \phi_t\}$, where T is called the time set, X is a state space (a metric space with metric d), A is the set of initial states $A \subset X$ and $\phi_t : X \to X$ is a family of evolution operators parameterized by $t \in T$ satisfying the following properties: for $x \in X, \phi_0(x) = x$, and $\phi_{t+s} = \phi_t \circ \phi_s$.

Remark 2. Note that in our definition of dynamical system, it is allowed to have, in general, more than one evolution operator.

Dynamical systems are classified based on the properties of T, X and ϕ_t. The time set T is it continuous or discrete?. The state space X is it finite or infinite?, is it continuous or discrete?. Is it X finite-dimensional or infinite-dimensional?. The map ϕ_t: Deterministic or stochastic?, autonomous or time-dependent?, invertible or not?, etc. Some examples are: Turing machines, finite state automata, continuous systems, discrete systems, discrete event systems and hybrid systems (to mention some).

When $T = \mathcal{R} = (-\infty, \infty)$, we speak of a continuous-time dynamical system, and when $T = \mathcal{N} = \{0, 1, 2, \cdots\}$ we speak of a discrete-time dynamical system. We will consider T equipped with the absolute value as a normed space i.e., $(T, |\cdot|)$.

A dynamical system is generally defined by one or more differential or difference equations There are other important classes of dynamical systems as those defined by continuous differential equations, functional differential equations, semi-groups, to mention some.

Remark 3. When dealing with continuous dynamical systems determined by ordinary differential equations on $\mathcal{R}^n$, we define the euclidean metric d as:

$$d(x,y) = \mid x - y \mid = \left[\sum_{1=1}^{n}(x_i - y_i)^2\right]^{\frac{1}{2}}, \forall x, y \in \mathcal{R}^n.$$

For discrete dynamical systems determined by difference equations, X equipped with the above euclidean metric defines a metric space.

Definition 4. A dynamical system is said to be computable if and only if it is family of evolution operators (also called its trajectories) are obtained as solutions of its mathematical model.

Postulate A. An analog algorithm is a dynamical system.

Definition 5. A vocabulary $\mathcal{V}$ is a finite collection of fixed-arity (possibly nullary) function symbols. We assume that $\mathcal{V}$ contains the scalar (nullary) function true. A first-order structure X of vocabulary $\mathcal{V}$ is a non-empty set S, the base set (domain) of X, together with interpretations of the function symbols in $\mathcal{V}$ over S, denoted by $\|f\|_{\mathcal{X}}$. Similarly, the interpretation of a term $f(t_1, \cdots, t_n)$ in X is recursively defined by $\|f(t_1, \cdots, t_n)\|_X = \|f\|_X(\|t_1\|_X, \cdots, \|t_n\|_X)$. Let X and Y be structures of the same vocabulary $\mathcal{V}$. An isomorphism from X onto Y is a one-to-one function ζ from the base set of X onto the base set of Y such that $f(\zeta t_1, \cdots, \zeta t_n)) = \zeta(t_0)$ in Y whenever $f(t_1, \cdots, t_n) = t_0$ in X.

Definition 6. A state transition system, consists of a set of states S, a subset I of initial states, transition functions on states, which determines the next-state relation, (states with no “next” state, will be terminal states).

Postulate B (Abstract state). States are first order structures with equality, sharing the same fixed, finite vocabulary. States and initial states are closed under isomorphism. Transitions preserve the domain, and transitions and isomorphisms commute.

Definition 7. An abstract transition system is a state transition system, whose function symbols f are interpreted as functions, and that satisfies postulate B, where the transition function on states is equal to ϕ_t.

Postulate C. An analog algorithm is an abstract transition system.

Definition 8. If f is a j-ary function symbol in vocabulary $\mathcal{V}$, and a is a j-tuple of elements of the base set of X, then the pair (f, a) (also denoted by $f(a)$) is called a location. We denote by $\|f(a)\|_X$ its interpretation in X. If (f, a) is a location of X then $(f, a, \|f(a)\|_X)$ is an update of X. When Y and X are structures over the same domain and vocabulary, $Y \setminus X$ denotes the set of updates $\triangle^+ = \{(f, a, \|f(a)\|_Y) : \|f(a)\|_Y \neq \|f(a)\|_X|\}$.

Definition 9. An infinitesimal generator is a function $\triangle$ that maps the state space X to a set $\triangle(X)$ of updates, and preserves isomorphisms i.e., if ζ is an isomorphism of states X, Y, then for all updates $(f, a, \|f(a)\|_X) \in \triangle(X)$, we have an isomorphic update $(f, \zeta a, \zeta\|f(a)\|_X) \in \triangle(Y)$.

Definition 10. A semantics ψ over a class C of sets S is a partial function mapping initial evolutions over some $S \in C$ to an element of S.

Definition 11. The infinitesimal generator associated with a semantics ψ maps the state space X, such that $\psi[X,f,a] = \psi(\|f(a)\|_{\phi_t(X)})$ is defined for all locations (f,a), to the set of updates $\triangle_\psi(X) = \{(f,a,\psi[X,f,a]) : (f,a) \in X\}$.

Remark 12. When $T = \mathcal{R}$, an example of semantics over the class of sets S containing T is the derivative ψ_{der}, when it exists. When $T = \mathcal{N}$, an example of semantics over the class of all sets would be the function $\psi_{\mathcal{N}}$ mapping f to $\psi_{\mathcal{N}}(f_n) = f_{n+1}, n \in \mathcal{N}$.

Remark 13. From now on, we assume that some semantics ψ is fixed to deal with different types of dynamical systems, it could be ψ_{der}, but it could also be another one. However,it is assumed that the class of dynamical systems is restricted to those that guarantee the existence of the respective semantics and as a result its associated set of updates is well defined. Therefore, not all possible dynamical systems are allowed.

Postulate D. For any analog algorithm, there exists a finite set T of variable free terms over the vocabulary $\mathcal{V}$, such that for all states X and Y that coincide for T, $\triangle_\psi(X) = \triangle_\psi(Y)$.

Definition 14. An abstract state machine, or ASM, is a state-transition system in which algebraic states (no predicate symbols) store the values of functions of the current state. Transitions are programmed using a convenient language based on guarded commands for updating individual states. ASM captures the notion that each step of an algorithm performs a bounded amount of work, whatever domain it operates over, so are central to the development.

An abstract state machine (ASM) is given by: a set S of algebraic states (no predicate symbols), closed under isomorphism, sharing a vocabulary $\mathcal{V}$, a set (or proper class) I of initial states, closed under isomorphism, and a program P, consisting of finitely many commands, each taking the form of a guarded assignment:

$$\text{if} \quad q \quad \text{then} \quad t := u,$$

for terms t and u over the vocabulary, and q is a conjunction of equalities and inequalities between terms i.e., given a state α which belongs to S, program P defines and therefore computes the following sub-set of the set of updates $\triangle^+$, $\{f(\|s\|_\alpha := \|u\|_\alpha : (\text{if} \quad p \quad \text{then} \quad f(s) := u) \quad \in P \quad and \quad \|q\|_\alpha = \quad true\}$.

In addition to the rule of the ASM program (see definition [14]) we introduce the following rules.

Definition 15. If each $R_1, R_2, \cdots, R_k$ are rules of the ASM i.e.,

$$\text{if} \quad q \quad \text{then} \quad t := u,$$

then:

$$\text{par} \quad R_1, R_2, \cdots, R_k \quad \text{endpar},$$

is a rule which executes them in parallel, with $\triangle_\psi(X)$ equal to the union of the same sub-set of updates given in definition [14] for each $R_1, R_2, \cdots, R_k$.

Definition 16. A rule of the form $Dynamic(f(t_1, t_2, \cdots, t_j), t_0)$ where f is a symbol of arity- j and, $t_0, t_1, t_2, \cdots, t_j$ are variable free terms, then the rule is defined by $\psi[X, f, (t_1, t_2, \cdots, t_j)] = \psi(f(t_1, t_2, \cdots, t_j)) := t_0$, where $\{\psi[X, f, (t_1, t_2, \cdots, t_j)]\}$ is an element of the set of updates $\triangle_\psi(X)$. In addition if $R_1, R_2, \cdots, R_k$ are rules of the form $Dynamic$ then:

$$\text{par} \quad R_1, R_2, \cdots, R_k \quad \text{endpar},$$

is also a rule, with $\triangle_\psi(X)$ being equal to the union of $\{(f_i, a_i, \psi[X, f_i, a_i]) : (f_i, a_i) \in X\}$ for $i = 1, \cdots, k$.

Definition 17. If ϕ is a boolean term and R_1 and R_2 are rules then, if ϕ then R_1 else R_2 is a rule.

The following result plays a fundamental role in the proof of the Church thesis for analog algorithms.

Theorem 18. For every algorithm of vocabulary $\mathcal{V}$, there is a program of the $\psi - ASM$, which for all states has the identical set of updates.

Example 19. Let us consider a simple pendulum whose dynamics is described by the following second order differential equation $\theta'' + \frac{g}{l}\theta = 0$. Its evolution is described by its associated set of updates (with ψ_{der}) of the following program rule:

$$\text{par} \quad Dynamic(\theta, \theta_1), Dynamic(\theta_1, -\frac{g}{l}\theta) \quad \text{endpar}.$$

Example 20. [5] A discrete event system, is a dynamical system whose state evolves in time by the occurrence of events at possibly irregular time intervals. Place-transitions Petri nets (commonly called Petri nets) are a graphical and mathematical modeling tool applicable to discrete event systems in order to represent its states evolution, whose mathematical model is given in terms of difference equations.

The matrix difference equation describing the dynamical behavior of a Petri net with m places and t transitions is represented as [4]:

$$M_{n+1} = M_n + A^T u_n, \quad (1)$$
$$n \in \mathcal{N}, M_n \in \mathcal{N}^m \text{and } u_n \in \{0,1\}^t.$$

This evolution is described by its associated set of updates (with $\psi_{\mathcal{N}}$) of the following program rule:

$$\begin{gathered} \text{par} \\ \textit{Dynamic}\left(M_n(p_1), M_n(p_1) + \sum_{j=1}^{t} a_{j1} u_n(j)\right), \\ \cdots, \\ \textit{Dynamic}\left(M_n(p_m), M_n(p_m) + \sum_{j=1}^{t} a_{jn} u_n(j)\right) \\ \text{endpar.} \end{gathered} \quad (2)$$

Notice that if M' can be reached from some other marking $M = M_n$ for some $n \in \mathcal{N}$ through a firing sequence $\{u_0, u_1, ..., u_{d-1}\}$writing equation (1) for each one of the elements of the firing sequence, and summing up, we obtain that:

$$M' = M + A^T u, \ u = \sum_{k=0}^{d-1} u_k. \quad (3)$$

Equation (3) would result in an ASM program, where the program rule (2) appears d times.

The proposed model can also adequately describe hybrid systems, made of alternating sequences of continuous evolution and discrete transitions.

Example 21. Let us consider a simple model of a bouncing ball, a classic example of a hybrid dynamical system, whose mathematical model is given by the equations $x^{''} = -gm$, where g is the gravitational constant and $v = x^{'}$ is the velocity, except that upon impact, each time $x = 0$, the velocity changes according to $v = -k \cdot v$; where k is the coefficient of impact. Every time the ball bounces, its speed is reduced by a factor k. Its evolution is described by its associated set of updates (with ψ_{der}) of the following program rule:

$$\text{if} \quad x = 0 \quad \text{then} \quad \text{if} \quad True \quad \text{then} \quad v := -k \cdot v$$

$$\text{else} \quad \text{par} \quad Dynamic(x, v), Dynamic(v, -gm) \quad \text{endpar.}$$

Definition 22. A $\psi - ASM$ comprises the following: an ASM program, a set S of first-order structures with equality over some finite vocabulary $\mathcal{V}$ closed under isomorphisms with a subset S_0 of S closed under isomorphisms, and a well defined update set of computations $\triangle_\psi$ associated with ψ.

Definition 23. An analog algorithm is a $\psi - ASM$ which satisfies postulates A, B, C and D.

We are assuming for that for each dynamical system, the trajectories can be computed from the description of its dynamical system. (As for example, in the case of nonlinear differential equation, the Lipschitz conditions are satisfied, etc). In other words not all dynamical systems are contemplated just those of them which guarantee their existence.

Definition 24. A semantics ψ is unambiguous if for all sets S of first-order structures over some finite vocabulary $\mathcal{V}$ closed under isomorphisms, and for all subsets $S_0 \in S$ closed under isomorphisms, whenever there exists some ϕ and a $\psi - ASM$, then ϕ is unique.

Theorem 25. Assuming ψ is unambiguous, for every process (algorithm) satisfying the postulates A, B, C and D, there is an equivalent $\psi - ASM$.

Theorem 26. (The Church thesis for analog algorithms) The dynamical system is computable if and only if the $\psi - ASM$ computes them.

If the dynamical system is computable (recall definition 33) there exists a procedure (algorithm) which computes its trajectories from its mathematical model description and therefore, the $\psi - ASM$ program will be able to emulate and compute these trajectories by a proper definition of its rules (see 18, 22). For the other side of the implication, given a $\psi - ASM$ program which first interprets the fixed dynamical system and then computes its trajectories, we define a numerical procedure which mimics it and therefore computes the dynamical system's trajectories. In fact, its trajectories define an exact mathematical model of themselves.

ISSN 2007-9737

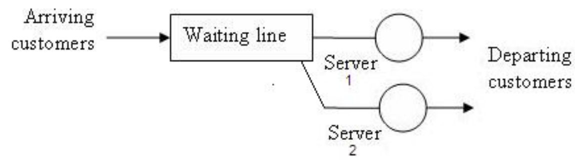

Fig. 1. Two server queuing system

2.1 Stability and Stabilization of Analog Algorithms

In this section, we will focus our attention to study the class of continuous and discrete-time dynamical systems defined by differential or difference equations, leaving other types for future work. We will begin by recalling some basic definitions in stability theory for this class [5, 7].

Definition 27. Stability

— Let us consider a dynamical system represented by the following differential equation:

$$\begin{aligned} dx/dt = &f(t,x) : x(0) = \\ &x_0 \in \mathcal{R}^n, x \in \mathcal{R}^n, \\ &f : \mathcal{R}^+ \times \mathcal{R}^n \to \mathcal{R}^n \text{continuous.} \end{aligned} \tag{4}$$

We say that state $x = 0$ of system (4) is stable if and only if, $\forall t_0 \in \mathcal{R}^+$ and $\forall \varepsilon > 0$ $\exists\ \delta = \delta(t_0, \varepsilon) > 0$ such that if $\| x_0 \| < \delta \Rightarrow \| x(t, t_0, x_0) \| < \varepsilon\ \forall t \in (t_0, \infty)$.

— Let us consider a dynamical system represented by the following difference equation:

$$\begin{aligned} x(n+1) = &f[n, x(n)] : x(n_o) = x_0, \\ &n \in \mathcal{N}_{n_0}, x(n) \in \mathcal{R}^n, \\ &f : N_{n_0} \times \mathcal{R}^n \to \mathcal{R}^n \text{continuous.} \end{aligned} \tag{5}$$

We say that state $x = 0$ of system (5) is stable if and only if, $\forall n_0 \in \mathcal{N}$ and $\forall \varepsilon > 0$ $\exists$ $\delta = \delta(n_0, \varepsilon) > 0$ such that if $\| x_0 \| < \delta \Rightarrow \| x(n, n_0, x_0) \| < \varepsilon\ \forall n \in N^+_{n_0}$.

Now, let us divide the set of structures i.e., the set of states of the dynamical system, in unstable and stable sets $X = \{X_{un}, X_s\}$.

Definition 28. An analog algorithm is said to be stable if and only if the dynamical system is stable if and only if the unstable structures are empty or they are not attained as the program of the $\psi - ASM$ executes.

A clear example of an unstable analog algorithm is the one defined for chaotic dynamical systems.

Let us suppose that it is possible to pass from unstable structures to stable structures by properly defining the rules of the $\psi - ASM$ program, then we will obtain a stable analog algorithm i.e., we have managed to stabilize the unstable algorithm i.e., the dynamical system is stabilizable.

Definition 29. An analog algorithm is said to be stabilizable if and only if it is possible to avoid the unstable structures by properly defining the rules of the $\psi - ASM$ program.

3 Discrete Event Dynamical Systems: A Case Study [5, 6]

A discrete event system, is a dynamical system whose state evolves in time by the occurrence of events at possibly irregular time intervals. Place-transitions Petri nets (commonly called Petri nets) are a graphical and mathematical modeling tool applicable to discrete event systems in order to represent its states evolution.

Timed Petri nets are an extension of Petri nets that model discrete event systems where now the timing at which the state changes is taken into consideration. One of the most important performance issues to be considered in a discrete event dynamical system is its stability. By proving stability one is allowed to preassigned the bound on the discrete event systems dynamics performance (the reader not familiar with these concepts is encouraged to see [4, 5, 6] and the references quoted therein).

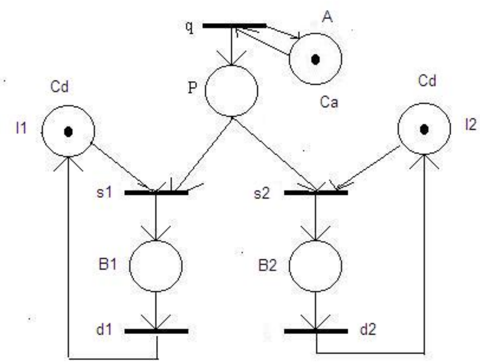

Fig. 2. Timed Petri net model

Consider a two server queuing system (Fig. 1.) whose timed Petri net (TPN) model is depicted in Fig. 2. Where the events (transitions) that drive the system are: q: customers arrive to the queue, s1, s2: service starts, d1,d2: the customer departs.

The places (that represent the states of the queue) are: A: customers arriving, P: the customers are waiting for service in the queue, B1, B2: the customer is being served, I1, I2: the servers are idle. The holding times associated to the places A and I1, I2 are Ca and Cd respectively, with $Ca > Cd$.

The PN (TPN) is unbounded since by the repeated firing of q, the marking in P grows indefinitely. However, employing Lyapunov and Max-Plus algebra techniques, it has been shown that by taking $u = [Ca, Ca/2, Ca/2, Ca/2, Ca/2]$, the PN is stabilizable which implies that the TPN is stable i.e., the load has to be equally divided between the two servers [6].

We have already discussed a $\psi - ASM$ whose program describes the dynamical behavior of a Petri net (see equation 20, equation (3)), therefore setting in our program $m = 6, t = 5$, and $u = [Ca, Ca/2, Ca/2, Ca/2, Ca/2]$, we are able to bound the marking in P or equivalently to avoid the unstable states of the queuing system i.e., the set X_{un} of our analog algorithm.

We conclude that by choosing properly the rules of the program $\psi - ASM$ the analog algorithm for the two server queuing system is stabilizable.

4 Stability of Analog Algorithms in Terms of Lyapunov Energy Functions for Analog Algorithms with Continuous and Discontinuous States

In this section, we consider the stability concept for analog algorithms with continuous and discontinuous states in terms of Lyapunov energy functions. The results presented in this section, become a generalization of what was discussed in sub-section 2.1 and section 3, and includes them as particular cases. It applies to many classes of discontinuous dynamical systems including hybrid systems and switched systems. We will deal with analog algorithms whose states are structures of vocabulary $\mathcal{V}$, where now the the base set S is a metric space (S, d), with metric d.

Definition 30. Let us consider an analog algorithm, we will say that the state X, with $a \in S$ and time-indexed location $f_{t,t_0}(a)$, where t and t_0 belong to T, is stable if and only if $\forall t_0 \in T$ and $\forall \varepsilon > 0 \; \exists \; \delta = \delta(t_0, \varepsilon) > 0$ such that if given $a' \in S$, with $d(a', a) < \delta \Rightarrow d(\|f_{t,t_0}(a')\|_X, \|f_{t,t_0}(a)\|_X) < \varepsilon$ $\forall t \in T$.

ISSN 2007-9737

Definition 31. Let us consider an analog algorithm, we will say that the state X, with $a \in S$ and time-indexed location $f_t(a)$ is continuous at $t \in T$, if and only if $\forall \varepsilon > 0 \; \exists \; \delta = \delta(t) > 0$ and a state Y such that if given $t' \in T$, with $\mid t - t' \mid < \delta \Rightarrow d(\|f_t(a)\|_X, \|f_{t'}(a)\|_Y) < \varepsilon$.

Definition 32. A continuous function $\alpha : [0, \infty) \rightarrow [0, \infty)$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and $\alpha(0) = 0$.

Postulate E. The Lyapunov energy function associated to the analog algorithm at its starting time point $t_0 \in T$ multiplied by some finite constant $C \geq 1$ bounds the whole Lyapunov energy function, transferred or transformed of the whole analog algorithm, as the Lyapunov energy function evolves in time.

Theorem 33.Let us consider an analog algorithm with the possibility of having discontinuous states at points $\{t_1, t_2, \cdots\} \subseteq T$. Assume there exists a Lyapunov function $V : S \times T \rightarrow \mathcal{R}^+$ and two functions α_1, α_2, which belong to $\mathcal{K}$, such that:

$$\alpha_1(d(\|f_{t,t_0}(a')\|_X, \|f_{t,t_0}(a)\|_X)) \leq V(\|f_{t,t_0}(a')\|_X, t)$$
$$\leq \alpha_2(d(\|f_{t,t_0}(a')\|_X, \|f_{t,t_0}(a)\|_X)),$$

for all $a, a' \in S$, $t, t_0 \in T$. Assume Postulate E and that $\|f_{t_0,t_0}(a')\|_X = a'$ holds, then the analog algorithm is stable.

We want to show that there exists a $\delta = \delta(t_0, \varepsilon) > 0$ such that given a' with $d(a', a) < \delta \Rightarrow d(\|f_{t,t_0}(a')\|_X, \|f_{t,t_0}(a)\|_X) < \varepsilon \; \forall t \in T$. Claim $\delta = \alpha_2^{-1}(C^{-1}\alpha_1(\varepsilon))$ does the job:

$$d(\|f_{t,t_0}(a')\|_X, \|f_{t,t_0}(a)\|_X) \leq \alpha_1^{-1}(V(\|f_{t,t_0}(a')\|_X, t))$$
$$\leq \alpha_1^{-1}(CV(\|f_{t_0,t_0}(a')\|_X, t_0)) = \alpha_1^{-1}(CV(a', t_0))$$
$$\leq \alpha_1^{-1}(C\alpha_2(d(a', a))) < \varepsilon.$$

Here postulate E has been used in the second inequality and the equation $\|f_{t_0,t_0}(a')\|_X = a'$ in the first equality.

It is worth mentioning that the preceding analysis applies to many classes of discontinuous dynamical systems, including hybrid systems and switched systems.

An example of a stable analog algorithm whose Lyapunov function satisfies the conditions imposed by theorem 33 is the one provided in [8], which consists of a ball in a constant gravitational field bouncing inelastically on a flat vibrating table. It is interesting to see how the Lyapunov function, proposed in the cited paper, monotonically decreases as t increases i.e., Postulate E holds with $C = 1$.

Consider the switched system defined by the following scalar differential equation:

$$dx/dt = \begin{cases} ln(2)x & \text{if } t \in [t_0 + 2k, t_0 + 2k + 1], \\ -ln(4)x & \text{if } t \in [t_0 + 2k + 1, t_0 + 2(k+1)], \end{cases}$$

where $k \in \mathcal{N}$, $x \in \mathcal{R}$, and $t_0 \in \mathcal{R}^+$. Then, taking $V(X) = \mid X \mid$ as our Lyapunov function, all the conditions of theorem 33 with $C = 2$ are satisfied, therefore concluding stability which implies that we get a stable analog algorithm.

5 Dynamical Systems with Non-Unique Trajectories

For continuous dynamical systems defined by differential ordinary differential equations, it is well-known that the continuity of the dynamical system does not guarantee uniqueness of solutions. Likewise, for discontinuous dynamical systems, uniqueness of solutions is not guaranteed in general, either no matter what notion of solution is chosen.

The lack of uniqueness of solutions generally requires a little bit of extra analysis because, we need to take into account the possibly multiple solutions starting from each initial condition.

This multiplicity leads us to consider the stability concept together with the adjectives *total* and *partial*. Roughly speaking, total is used when the stability property is satisfied for all solutions starting from each initial condition.

On the other hand, partial is used when the stability property is satisfied by at least solutions starting from each initial condition. Locally Lipschitz is the most common requirement invoked to guarantee uniqueness of solution.

Example 36. Consider the dynamical system defined by the following ordinary differential equation:

$$x^{'}(t) = \Big[\mid x \mid \Big]^{\frac{1}{2}}, x \in \mathcal{R},$$

which is continuous everywhere, and locally Lipschitz on $\mathcal{R}\setminus\{0\}$. This differential equation has two solutions starting from 0, namely: $\phi_1 : [0,\infty) \to \mathcal{R}$, $\phi_{t_1} = 0$, and $\phi_2 : [0,\infty) \to \mathcal{R}$, $\phi_{t_2} = \frac{t^2}{4}$.

Definition 37. An analog algorithm for non unique evolution operators is the Cartesian product of analog algorithms, where each one of the analog algorithms that belong to the Cartesian product satisfy all what was discussed in section 2.

Postulate E. An analog algorithm for non unique evolution operators is a dynamical system $\{T, X, A, \{\phi_{t_i}\}_{i\in\mathcal{N}}\}$, where T, X, A are defined as in (2), $\{\phi_{t_i}\}_{i\in\mathcal{N}}$ is an indexed family of evolution operators parametrized by T, and for each $i \in \mathcal{N}$ fixed, there corresponds one member of the Cartesian product of analog algorithms.

Remark 38. From definition (37) by taking Cartesian products, it is immediate to generalize all the properties of analog algorithms, (given in section 2), to analog algorithms for non unique evolution operators, where now we have an ASM program associated to each one of the evolution operators and which defines the ASM program of the analog algorithm for non unique evolution operators.

Continuing with our previous example, the ASM program of the analog algorithm for non unique evolution operators that models it, turns out to be composed by the following two ASM programs:

— $\text{if} \quad True \quad \text{then} \quad \phi_{t_1} := 0,$

— $\text{if} \quad True \quad \text{then} \quad \phi_{t_2} := \frac{t_2^2}{4}.$

The set of updates is given by $\triangle_\psi(X) = \{(\phi_{t_1} := 0), (\phi_{t_2} := \frac{t_2^2}{4})\}$. Even more, the analog algorithm for non unique evolution operators results to be partially stable.

References

1. **Bournez, O., Dershowitz, N., Neron, P. (2016).** Axiomatizing Analog Algorithms.
2. **Gurevich, Y. (2000).** Sequential abstract-state machines capture sequential algorithms. ACM Trans Comput Log, Vol. 1.
3. **Retchkiman, K.Z., Dershowitz, N. (2019)**. The Church thesis, its proof, and the notion of stability and stabilization for analog algorithms. Communications in Applied Analysis, Vol. 23, No. 2.
4. **Murata, T. (1989)**. Petri nets: Properties, analysis, and applications. Proceedings of the IEEE, Vol. 77, pp. 541–580.
5. **Retchkiman, Z. (2005)**. Stability theory for a class of dynamical systems modeled with Petri nets. International Journal of Hybrid Systems, Vol. 4, No. 1.
6. **Retchkiman, Z. (2012)**. Timed Petri Net Modeling and Lyapunov/Max-Plus-Algebra Stability Analysis for a type of Queuing Systems by means of timed Petri nets, Lyapunov methods and max-plus algebra. International Journal of Pure and Applied Mathematics, Vol. 77 No. 3.
7. **Lakshmikantham, V., Matrosov, V.M., Sivasundaram, S. (1991)**. Vector Lyapunov functions and stability analysis of nonlinear systems. Kluwer Academic Publ.
8. **Heimsch, T.F., Leine, R.I. (2011)**. A novel Lyapunov-like method for the non-autonomous bouncing ball system. 7th European Nonlinear Dynamics Conference (ENOC).

Article received on 06/05/2019; accepted on 20/04/2020.
Corresponding author is Zvi Retchkiman Konigsberg.