On The Causality of Mixed-Signal and Hybrid Models

Jie Liu[1] and Edward A. Lee[2]

[1] Palo Alto Research Center,
Palo Alto, CA 94304
jieliu@parc.com
[2] Department of EECS, University of California
Berkeley, CA 94720
eal@eecs.berkeley.edu

Abstract. This paper extends the application of the Cantor metric as a mathematical tool for defining causalities from pure discrete models to mixed-signal and hybrid models. Using the Cantor metric, which maps timed signals, continuous or discrete, into a metric space, we define causality as contractive properties of processes operating on these signals. Thus, the Banach fixed point theorem can be applies to establish conditions for the existence, uniqueness, and liveness of the behaviors for mixed-signal and hybrid systems. The results also provide theoretical foundations for the simulation technologies for such systems, including the time-marching strategy, evaluation of feedback loops, and the necessity of supporting rollback.

1 Introduction

Engineering systems that exhibit both continuous and discrete dynamics have obtained great attention from many perspectives, such as modeling, simulation, control, and verification. Although continuous-time models and various discrete models themselves are relatively well-understood, the integration of different models imposes new questions on system properties such as definability (existence of behaviors), determinism (uniqueness of the behavior) and liveness (the behavior extends to time ∞). In the context of hybrid automata (e.g. [1], [2]), these questions have been analyzed from a state trajectory point of view [3], [4], [5]. However, due to the explicit representation of continuous and discrete states, the compositions (I/O composition in particular) of hybrid automata can be quite involved, which makes state based analysis techniques not quite scalable to complex systems [6].

Mixed-signal models, on the other hand, characterize that the signals connecting different components of a system may be continuous or discrete. Hybrid automata can be examples for such components. Mixed-signal models hide the implementation detail of each component, thus are widely used as a coordination model in modeling languages and simulation tools, such as VHDL-AMS [7] and Simulink [8].

This paper takes a denotational approach and studies the existence, uniqueness, and liveness properties of mixed-signal models in a tagged-signal semantic framework [9], under notions of various causalities defined using the Cantor metric. This framework allows us to apply the Banach fixed point theorem to define the denotational behavior for mixed-signal systems. The strength of this approach is its generality: causalities are defined based on the input and output signals rather than on the implementation of the components, which makes the analysis applicable to a wide variety of models, including pure discrete event models [10], pure continuous-time models, hybrid automata, and practically all timed systems.

A practical implication of the discussion on causalities is the simulation strategies of mixed-signal and hybrid systems. By introducing a notion of ideal ODE solvers, we abstract the simulation of continuous-time systems into a sequence of discrete operations. This discrete abstraction allow us to apply the Banach fixed point iteration to show that the commonly used time-marching simulation strategy is compatible with the denotational semantics, and it yields a correct behavior if there exists one.

The rest of the paper is organized as the following. Section 2 gives an overview of the tagged-signal model, with a focus on a formal definition of mixed-signal processes. Section 3 defines three kinds of causalities and analyzes some typical mixed-signal processes in terms of their causality properties. It gives a sufficient condition for the existence, uniqueness, and liveness of the behaviors of mixed-signal systems using the Banach fixed-point theorem. Section 4 applies the causality concepts in the simulation of mixed-signal and hybrid systems, and shows the rationale of the common simulation strategies.

2 Tagged-Signal Model

The tagged-signal model is a denotational semantics framework for a variety of models of computation [9]. The model looks at the *signals* communicating among a set of components (called *processes*), and defines the behaviors of the processes as the set of signals they constrain.

2.1 Tags and Signals

In the tagged-signal model, an *event* $e = (t, v)$ is a tag-value pair. That is, e is a member of the set $E = T \times V$, where T is a set of tags, and V is a set of values. The set T can be finite, countably infinite, or uncountable; it may also be partially ordered or totally ordered. Here, the *ordering relation* is a reflexive, transitive, and antisymmetric relation, denoted by $\leq$. The order defined on T also defines the order of events in E.

A *signal* s is a set of events. That is, $s \in \wp(E)$, where $\wp(E)$ is the power set (i.e. the set of all subsets) of E. We denote by $S = \wp(E)$ the set of all signals, and by S^N a N tuple of signals. A signal s is called *functional* or *proper* if it

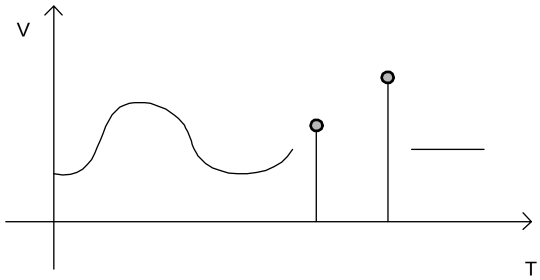

Fig. 1. This is neither a continuous-time signal nor a discrete-event signal.

is a (possibly partial[3]) function from T to V. That is, if $e_1 = (t, v_1) \in s$ and $e_2 = (t, v_2) \in s$, then $v_1 = v_2$. A proper signal can be written as a function $s : T \to V$, such that if $e = (t, v) \in s$, then $s(t) = v$. A signal s is called *totally ordered* or *ordinary* if it is totally ordered with respect to the ordering relation defined on E. In this paper, we only consider ordinary proper signals.

In the context of mixed-signal and hybrid system modeling, T represents time. More precisely, $T \subseteq R_0^+$, where R_0^+ is the set of all positive real numbers and zero, and T inherits the total ordering and metric from R_0^+. For $T_1 \subseteq T_2 \subseteq T$, T_1 is called the *prefix* of T_2, if for all $t \in T_1$ and for all $t' \in T_2$ but $t' \notin T_1$, the relation $t \leq t'$ holds. If $s_1 : T_1 \to V, s_2 : T_2 \to V$ and $s_1(t) = s_2(t), \forall t \in T_1$, then s_1 is called the prefix of s_2, written as $s_1 \sqsubseteq s_2$. In addition, we introduce an empty value λ to the value set, i.e. $V = R \cup \{\lambda\}$. A metric is not required for the value set. We denote by Λ a signal that contains only empty events, and Λ_N a N-tuple of such signals.

Under these definitions, the difference between various kinds of signals is captured in the topologies of the tag and value sets. A *continuous-time signal* has the entire R_0^+ as its tags and the real numbers R as the value set. A *partial continuous-time signal* is a function defined only on a *connected* prefix of R_0^+. The prefix may be open or closed. In a degenerate case, an event defined on a single point $\{0\}$ is a partial continuous-time signal.

Intuitively, a discrete-event signal only takes non-empty values at a "discrete" subset of T. Formally, a set $T_d \subset T$ is *discrete* if it is *order isomorphic* to a subset of integers [10]. That is, there exists a bijective map between the tags and a subset of integers that preserves the order. By introducing the empty value, a discrete-event signal is defined on the entire tag set R_0^+. A signal s is a *discrete-event signal* if there exists a discrete subset $T_d \subset R_0^+$ such that $\forall t \notin T_d, s(t) = \lambda$. Note that the signal shown in Figure1 is neither a continuous-time signal nor a discrete-event signal.

A continuous-time signal is not necessarily a *continuous signal*, which, in addition to having a connected tag set, is also a *continuous function* from T to

[3] A partial function is a function defined only on a subset of its domain.

R. A signal s on T is *piecewise continuous*, if there exist a discrete set $T_d \subset T$, such that s is continuous on $T - T_d$ and right continuous on T_d.

2.2 Continuous, Discrete, and Mixed-Signal Processes

From a denotational point of view, a *process* P is a subset of S^N for some N. A particular signal tuple $\mathbf{s} \in S^N$ is said to *satisfy* the process if $\mathbf{s} \in P$. Thus, a process is a set of possible behaviors. Intuitively, the implementation of a process P has N *ports* and S^N are all possible signals on these ports. It is useful in the context of this paper to have the notions of inputs and outputs of processes, depending on whether the process constrains the signal. An input to a process $P \subseteq S^N$ is an external constraint $A \subseteq S^N$ such that $A \cap P$ is the total set of acceptable behaviors under that input. A process is *functional* if the output signals are given as a function of the input signals.

Viewing processes as sets of signal is a powerful concept, such that the composition of processes are simply reordering and projections of signal tuples, and the composed behaviors are the intersection of component process behaviors. A composition of processes is *definable* if the behavior intersection is not empty; and it is *deterministic* if the behavior intersection has exactly zero or one element. Figure 2 shows the serial, serial/parallel, parallel, and feedback compositions of two processes P_1 and P_2. Connecting the output of one process to the input of another imposes a constraint that the two signal to be the same. A full discussion of process composition is out of the scope of this paper, and can be found in [9].

We further distinguish the types of processes by the signals they contain. A process $P \subseteq S^N$ is *piecewise-continuous* (or a *continuous process*, in short) if all S in S^N are (possibly partial) piecewise-continuous signals. Similarly, P is a *discrete-event process* (or, a *discrete process*, in short) if all S in the tuple are discrete-event signals. Some processes may contain both piecewise-continuous and discrete-event signals in their behavior. Such processes are called *mixed-signal processes*. A mixed-signal system is a composition of processes in S^N, such that there exists N_C, N_D, and $S^N = S^{N_C} \times S^{N_D}$, where S^{N_C} is a tuple of piecewise-continuous signals and S^{N_D} is a tuple of discrete-event signals.

For example, an integrator is a continuous process. It takes an integrable signal as input and produces an integrable output signal whose derivative is the input signal almost everywhere. An ordinary differential system (ODS),

$$\dot{x} = f(x, u, t) \quad (1)$$

$$y = g(x, u, t) \quad (2)$$

$$x(0) = x_0 \quad (3)$$

is also a continuous process, where u is the input signal, y is the output signal, and x is the state variable. There are many examples of discrete processes. A time delay process delays all events in the input signal with a specific amount of time. An I/O automaton can be viewed as a discrete process that does not introduce delays from input events to output events. Timed I/O automata, however, are discrete processes that manipulate both values and tags in the signals.

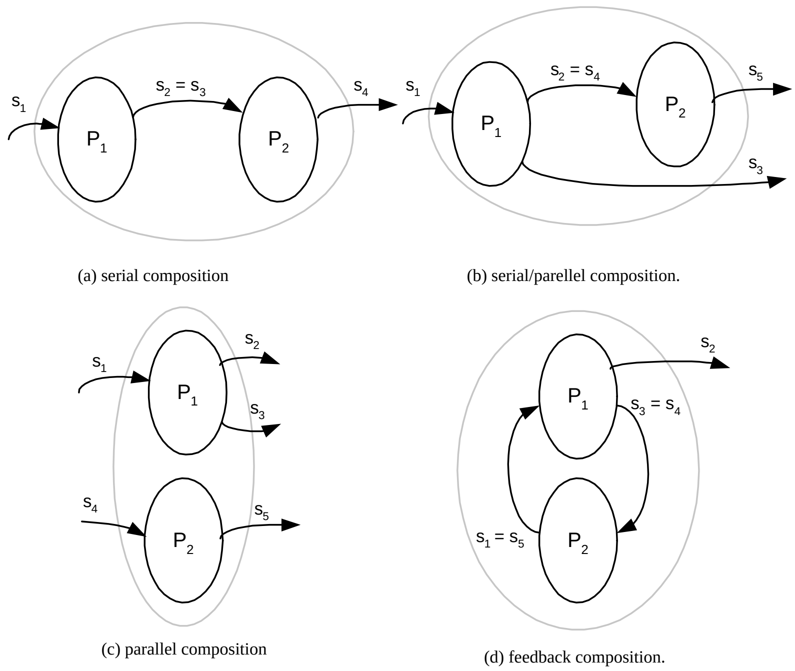

Fig. 2. Two processes P_1, P_2 and their compositions. I/O composition imposes additional constraints on the possible behaviors of the processes, in the sense that the signals on the connected arcs have to be the same.

Event generators are processes that have at least one continuous-time input signal and discrete-event output signals. Waveform generators converts discrete-event input signals to (usually piecewise) continuous output signals. They are examples of mixed-signal processes. Hybrid automata, in their most general forms, can have both piecewise-continuous and discrete-event inputs and outputs, thus are mixed-signal processes.

Notice that these definitions classify continuous and discrete processes by the signals they exhibit, rather than their internal implementation. A discrete process may be implemented internally by continuous-time differential equations with event detection and waveform generation mechanisms. A hybrid automata may only expose piecewise-continuous signals at its interface, in which case, it is a continuous process.

3 Causality

Using the tagged-signal model allows us to study system behaviors as signals in a signal space, in particular, a signal space with metric.

3.1 Cantor Metric and Causality

The Cantor metric is a metric that compares the distance among timed signals. Consider a N tuple of (mixed) signals S^N defined on the tag set $T \subseteq R_0^+$. For $\mathbf{s}, \mathbf{s}' \in S^N$, $\mathbf{s}(t) = [(t, v_1), ..., (t, v_N)]^\top$ and $\mathbf{s}'(t) = [(t, v_1'), ..., (t, v_N')]^\top$, we say that $\mathbf{s}(t) \neq \mathbf{s}'(t)$ if $\exists i \in \{1, ...N\}$, s.t., $v_i \neq v_i'$.

Definition 1. *For two signals* $\mathbf{s}, \mathbf{s}' \in S^N$*, the* Cantor metric *defines the distance between* $\mathbf{s}, \mathbf{s}'$ *as:*

$$d(\mathbf{s}, \mathbf{s}') = \sup\{\frac{1}{2^t} | \mathbf{s}(t) \neq \mathbf{s}'(t)\} \tag{4}$$

It is easy to check that this is indeed a metric, satisfying $d(\mathbf{s}, \mathbf{s}') \geq 0$, $d(\mathbf{s}, \mathbf{s}) = 0$, $d(\mathbf{s}, \mathbf{s}') = d(\mathbf{s}', \mathbf{s})$, and the triangle inequality. In fact, it is an *ultrametric*, satisfying a stronger form of the triangle inequality:

$$d(\mathbf{s}, \mathbf{s}'') \leq \max(d(\mathbf{s}, \mathbf{s}'), d(\mathbf{s}', \mathbf{s}'')). \tag{5}$$

Under this metric, two signals are *close* if they agree over a great amount of time. It is also easy to verify that under the Cantor metric, the space of mixed signals is *complete*, *i.e.* all Cauchy sequences of mixed signals converge to a mixed signal.

We use the Cantor metric to define three increasingly stronger notions of causality on functional processes:

Definition 2. *A functional process* $P : S^I \to S^O$ *is* causal *if for all* $\mathbf{s}, \mathbf{s}' \in S^I$,

$$d(P(\mathbf{s}), P(\mathbf{s}')) \leq d(\mathbf{s}, \mathbf{s}'). \tag{6}$$

Definition 3. *A functional process* $P : S^I \to S^O$ *is* strictly causal *if for all* $\mathbf{s}, \mathbf{s}' \in S^I$,

$$d(P(\mathbf{s}), P(\mathbf{s}')) < d(\mathbf{s}, \mathbf{s}'). \tag{7}$$

Definition 4. *A functional process* $P : S^I \to S^O$ *is* δ-causal *if for all* $\mathbf{s}, \mathbf{s}' \in S^I$, *there exists* $0 \leq \delta < 1$, *s.t.*

$$d(P(\mathbf{s}), P(\mathbf{s}')) \leq \delta \cdot d(\mathbf{s}, \mathbf{s}'). \tag{8}$$

Note that these definitions are compatible with but more precise than the common understanding of causality, which is usually stated as "the output of a process at time t should not depend on inputs that are later than t."

3.2 Causality of Continuous, Discrete, and Mixed-Signal Processes

In this section, we give some examples of continuous, discrete, and mixed-signal processes, and analyze their causality properties. Unless otherwise stated, throughout this section, we assume that P is a functional process $P : S^I \to S^O$ and $\mathbf{s}, \mathbf{s}' \in S^I$.

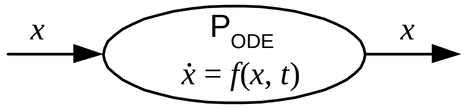

Fig. 3. An ordinary differential equation as a process P_{ODE}.

Memoryless Processes Memoryless processes are point-wise operators on input signals. That is, if P is memoryless and $\mathbf{s}(t) = \mathbf{s}'(t)$, then $P(\mathbf{s}(t)) = P(\mathbf{s}'(t))$, regardless of the other events in $\mathbf{s}$ and $\mathbf{s}'$. Thus, a memoryless process is causal but usually not strictly causal or δ-causal.

Integrators An integrator is a continuous process that takes integrable piecewise-continuous signals as input and produces a continuous signal as output. Let u be an input signal and x be the corresponding output signal, then an integrator implements:

$$x(t) = x(0) + \int_0^t u(\tau) d\tau \tag{9}$$

In the theory of Lebesgue integration (see e.g. [11]), for any $t \in T$, $u(t)$ has measure 0, thus, $x(t)$ depends only on $u[0,t)$ but not on $u(t)$. For two inputs u and u', satisfying $u(\tau) = u'(\tau), \forall \tau \in [0,t)$, the outputs of the integrator satisfies $x(\tau) = x'(\tau), \forall \tau \in [0,t]$, even when $u(\tau) \neq u'(\tau)$. However, this useful insight does not directly improve the causality of the integrator. In fact, an integrator is not strictly causal, since $d(x, x') = d(u, u')$.

Ordinary Differential Equations An ordinary differential equation (ODE)

$$\dot{x} = f(x,t), \; x(0) = x_0 \tag{10}$$

can be viewed as a process P_{ODE} mapping X, the set of all partial and complete solutions of (10), to X, as shown in Figure 3. Formally, $X = \{x[0, t_f] | t_f \in T$, and x satisfies the ODE in interval $[0, t_f]\}$. We define that the degenerate partial continuous-time signal $(0, x_0) \in X$. So, X is never empty.

Let M be the dimension of x, and $x, x' \in X$ be two inputs to P_{ODE} with $d(x, x') = 1/2^\tau$, then, by continuity, $x(\tau) = x'(\tau)$. The local existence and uniqueness theorem of ODE (see, e.g. [12]) states that if there exists $\sigma > \tau$ and $L, r > 0$, such that $f(x,t)$ satisfies the local Lipschitz condition:

$$\| f(u,t) - f(v,t) \| \leq L \| u - v \| , \tag{11}$$

for all $u, v \in \{z \in R^M | \; \|z - x(\tau)\| \leq r\}$ and for all $t \in [\tau, \sigma]$, then, there exists $\tau < \omega \leq \sigma$ such that the ODE (10) has a unique solution on $[\tau, \omega]$. That is, P_{ODE} extends the agreement of the partial solution by $\omega - \tau$ amount, *i.e.* P_{ODE} is strictly causal under the local Lipschitz condition with $d(P_{ODE}(x), P_{ODE}(x')) = 1/2^\omega < d(x, x')$.

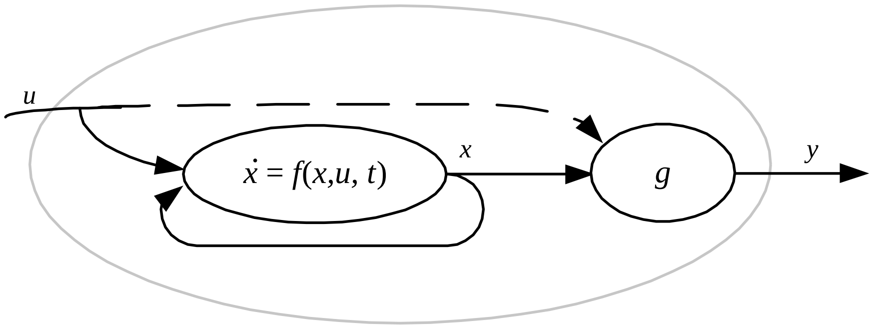

Fig. 4. An ordinary differential system.

In addition, if the ODE satisfies global Lipschitz condition, such that there is a smallest $\Delta > 0$ and P_{ODE} extends the solution for at least Δ amount in time for any inputs, then P_{ODE} is δ-causal, with $\delta = 1/2^{\Delta}$. The extension from the local solution to the global solution will be further discussed in section 3.3.

An ODS (1)-(3) can be viewed as a composition of an ODE process with input and a memoryless output map process, g, as shown in Figure 4.

If u is piecewise continuous and f is globally Lipschitz on its first argument, the existence and uniqueness theorem of ODE guarantees that the ODE (1) and (3) has a unique solution. It is causal from u to x. Similar to the integrator case, $x(t)$ does not depend on $u(t)$, for any $t \in R_0^+$. But this does not directly improve the causality. The memoryless output map (2) is also causal. Thus, an ordinary differential system is causal under the global Lipschitz condition. Note that if the output map g does not have u as its direct input, then $y(t)$ does not directly depend on $u(t)$.

Time-Event Generators Time-event generators take piecewise-continuous input signals and generate discrete events at a *predefined* set of discrete time instants. Given a discrete set of time points $T_d = \{t_1, t_2, ...\} \subset T$, a time-event generator $P_{TEG} : S^I \to S^O$ is a process that for a piecewise-continuous input $\mathbf{s} \in S^I$,

$$P_{TEG}(\mathbf{s})(t) = \begin{cases} G(\mathbf{s}[0, T_d]) & \text{if } t \in T_d \\ \lambda & \text{otherwise} \end{cases} \quad (12)$$

where $G(\mathbf{s}[0, T_d])$ is a function of the input signal up to time T_d.

Typically, a time-event generator omits some values in the input signal, and replace them with the empty value λ. Take a periodic sampler as an example, where T_d contains a set of equidistance points. Let ts be the sampling period. Suppose that for two inputs $\mathbf{s}, \mathbf{s}' \in S^I$, $d(\mathbf{s}, \mathbf{s}') = 1/2^{\tau}$, we examine the distance of the output signals. Let $\lceil \tau \rceil$ be the smallest element in T_d that is greater than or equal to τ. There are three cases:

1) if $\tau \notin T_d$, then $d(P_{TEG}(\mathbf{s}), P_{TEG}(\mathbf{s}')) \leq 1/2^{\lceil \tau \rceil} < 1/2^{\tau}$, strictly causal.

2) if $\tau \in T_d$ and $\mathbf{s}(\tau) = \mathbf{s}'(\tau)$, then $d(P_{TEG}(\mathbf{s}), P_{TEG}(\mathbf{s}')) \leq 1/2^{(\tau+ts)} < 1/2^{\tau}$, strictly causal.
3) if $\tau \in T_d$ and $\mathbf{s}(\tau) \neq \mathbf{s}'(\tau)$, then $d(P_{TEG}(\mathbf{s}), P_{TEG}(\mathbf{s}')) = 1/2^{\tau}$, causal.

Thus, unless the generator samples right on a discontinuous point of the piecewise-continuous input signal, it is strictly causal.

State-Event Generator Unlike time-event generators, a state-event generator produces an event if the trajectory of the piecewise-continuous input satisfies certain conditions. Typically, an output event is associated with a condition h and a value assignment rule r. For an input signal $\mathbf{s} \in S^I$, a condition $h(\mathbf{s}) = 0$ defines a surface in the value space R^I. A discrete event $e = (\tau, v)$ is in the output of a state-event generator if $h(\mathbf{s}(\tau)) = 0$ and there exists a nonempty open interval (τ', τ) such that $h(\mathbf{s}(t)) \neq 0$, for $t \in (\tau', \tau)$. The assignment r defines the value of e, *i.e.* $v = r(\mathbf{s}[0, \tau])$. We call this type of event *zero-reaching event.* Similar to time-event generators, if at the event occurrence time the input signal $\mathbf{s}$ is continuous, the process is strictly causal. Otherwise, the process is simply causal.

Sometimes, it is useful to specify an event condition that also takes the future trajectory into account. For example, a *transverse event* requires that the input not stay on the surface $h(\mathbf{s}) = 0$ after reaching the surface [13]. That is, there also exists an open interval (τ, τ''), s.t. $h(\mathbf{s}(t)) \neq 0$, for $t \in (\tau, \tau'')$. This includes *zero-crossing events* which require that the signal $\mathbf{s}$ be on two different sides of the surface before and after the event occurrence. It also includes *zero-touching events* which require that the signal $\mathbf{s}$ be on the same side of the surface before and after the event occurrence. Although transverse events may seem non-causal under this description, there are ways to define them using zero-reaching condition and the Lie derivatives if at the event occurrence point the input signal and the event surface are analytic [13].

Zero-Order Hold A zero-order hold (ZOH) process is one of the most primitive type of waveform generators. Given a discrete-event signal $s_d = \{(t_1, v_1), (t_2, v_2), ...\}$, and an initial value v_0, a zero-order hold process outputs a unique continuous-time extension of s_d, denoted by $zoh\langle s_d, v_0 \rangle$, such that $zoh\langle s_d, v_0 \rangle(t) = v_i$, for $t_i \leq t < t_{i+1}$ and $t_0 = 0$. Obviously, $zoh\langle s_d, v_0 \rangle$ is a piecewise-continuous signal, and a zero-order hold process is causal but not strictly causal.

Sampled Differential Systems A sampled differential system, as studied in discrete-time systems, is a composition of ZOH processes, ordinary differential systems, and periodic samplers, as shown in Figure 5. Here we assume that the input v and the output z have the same set of tags T_d, and that u does not feed directly into the output map g.

Let t_s be the sampling period, and $0 \in T_d$. We order the elements in $T_d = \{t_0, t_1, ...\}$ such that $t_0 = 0$ and $t_{k+1} - t_k = t_s$ for $k \geq 0$. Let $v, v' \in S^I$ be two input signals to the sampled differential system, and assume $d(v, v') = 1/2^{\tau} \neq 0$.

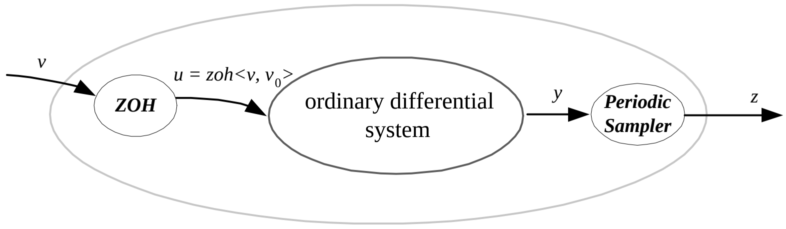

Fig. 5. A sampled differential system provides a discrete interface at input v and output z. The ordinary differential system process may be internally implemented by processes shown in Figure 4.

Then, there must exist some index k such that $\tau = t_k \in T_d$. Since the ZOH process is causal, $d(u, u') = 1/2^{\tau}$. However, for a differential system, $x(\tau) = x'(\tau)$ and $y(\tau) = y'(\tau)$, even though $u(\tau) \neq u'(\tau)$. As the periodic sampler samples at τ, the output $z(\tau) = z'(\tau)$. This equality will hold until the next sampling time t_{k+1}, i.e. $d(z, z') = 1/2^{t_{k+1}}$. Thus, from the input/output point of view, a sampled differential system is δ-causal, with $\delta = 1/2^{t_s}$.

Hybrid Automata In the formalism of hybrid automata [1], there is a set of discrete states, Σ, a set of continuous state variables X. At each discrete state in Σ, the automaton is refined into an ordinary differential system on some state variables in X. There are transitions among the discrete states. A hybrid automaton may have both continuous and discrete inputs and outputs. A discrete state σ may have *invariants* that specify the condition that the system can stay in σ. If a invariant is violated, a discrete transition must be taken. A transition may have *guards* and *actions*. The guards may depend on the discrete and continuous input signal values and specify the conditions that the transition may be taken. The actions is performed when the transition is taken, and may include producing discrete events and reseting the values of the continuous state variables in the destination discrete state.

Notice that in a hybrid automaton model, there is no mechanism to directly specify time delays from input events to output events. A transition is instantaneous. If a reachable transition is triggered by an input event, and the corresponding action produces an output event, then the hybrid automaton is causal. The only way to introduce strict causality in a hybrid automaton is to ensure that the automaton stays in a discrete state for some of time, so that the differential system that refines the discrete state becomes effective, similarly for δ-causality.

3.3 Existence, Uniqueness, and Liveness

Causality plays a central role in the existence, uniqueness, and liveness of behaviors of mixed-signal systems.

It is easy to verify that acyclic I/O compositions (e.g. cases (a), (b), and (c) in Figure 2) of functional processes are functional, and preserve causality. We have also shown in the case of sampled differential systems that a composition of causal and strictly causal processes — in this case, a ZOH process, an ODS process, and a periodic sampler process — may have a stronger causality than the individual processes. Functional processes have the property that given any input signal in the domain of the process, there is exactly one output signal. So, for acyclic compositions of functional processes with deterministic input signals, a behavior exists and is unique.

Feedback compositions are more complicated. Through sorting and projection of signals in the signal tuples S^N, a mixed-signal system with feedback can be viewed as a function $F : S^N \rightarrow S^N$. It is not obvious whether there exists any $\mathbf{s} \in S^N$, such that $F(\mathbf{s}) = \mathbf{s}$ (*existence*); if such $\mathbf{s}$ exists, whether it is unique (*uniqueness*); and whether the signal is defined on the entire R_0^+ (*liveness*). One example of mixed-signal systems lacking the liveness property is the *Zeno phenomena* where in a finite time interval there can be an infinite number of discrete events.

From the definitions, the forms of causality are "contraction" relations among input and output signals in a metric space, thus the Banach fixed point theorem may ensure that a system with feedback loops has a unique behavior under certain conditions. The Banach fixed point theorem states that for S^N complete, (which is true for mixed-signal systems), if F is δ-causal, then there is exactly one $\mathbf{s} \in S^N$ such that $F(\mathbf{s}) = \mathbf{s}$. This signal is called a *fixed point*. Moreover, the theorem also gives a constructive algorithm to find the fixed point. Given $\mathbf{s_0}$ in the domain of F, $\mathbf{s}$ is the limit of the sequence:

$$\mathbf{s_1} = F(\mathbf{s_0}), \mathbf{s_2} = F(\mathbf{s_1}), \mathbf{s_3} = F(\mathbf{s_2}), ... \tag{13}$$

Thus, the theorem gives a sufficient condition for existence and uniqueness of the behavior of a mixed-signal system.

A direct application of the theorem to ordinary differential equations is the obtainment of global unique solution under the global Lipschitz condition. Starting with $\mathbf{s_0} = \{(0, x_0)\}$, a partial continuous signal in the solution space, each application of the Banach fixed-point iteration corresponds to extending the local solution for at least Δ amount of time into the future:

$$\mathbf{s_0} \sqsubseteq \mathbf{s_1} \sqsubseteq \mathbf{s_2} \sqsubseteq ... \tag{14}$$

Since the sequence of solutions converges, it must embeds a Cauchy sequence, *i.e.* $d(\mathbf{s_M}, \mathbf{s_K}) \rightarrow 0$ as $M > K \rightarrow \infty$. Thus, the solution is unique on $[0, \infty)$. Similar analysis can be applied to mixed-signal systems, where the initial signal is $\Lambda_{N_D} \times \{(0, x_0)\}$, i.e. empty discrete events and the initial values for continuous state variables.

The δ-causality requirement is fairly strong. A closely related theorem (see e.g. [14], chapter 4) states that if F is strictly causal and S^N is complete, then there is at most one fixed point for F. Thus, strict causality guarantees determinism, but does not ensure that a feedback system has a behavior, nor is it

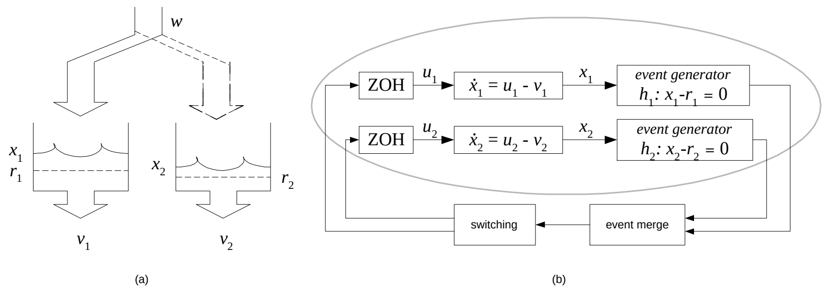

Fig. 6. The two-tank problem and its mixed-signal model.

enough to prevent Zeno phenomena. When we weaken the condition further, the simple causality does not even provide determinism.

The δ-causality requirement is also tight, in the sense that violation of this condition may introduce Zeno phenomena or non-determinism. For example, the two-tank problem is a classical hybrid system with a Zeno execution [15]. As shown in Figure 6 (a), x_i denotes the water level in tank $i \in \{1, 2\}$, and $v_i > 0$ is the constant flow of water out of tank i. Let w be the constant flow of water into the system, delegated exclusively to either tank 1 or tank 2, controlled by a switch. Let r_i be the reference level of tank i, such that if $x_i = v_i$, an alarm will be generated requesting the in-flow water to be switched to tank i. The switching logic is that whenever it receives an alarm, the in-flow water is directed to the requested water tank instantaneously. We further assume that $x_i(0) > r_i$, for $i = 1, 2$, that the in-flow water rate satisfies $v_i < w < v_1 + v_2$, and that it is initially directed to tank 1. A mixed-signal model of this system is shown in Figure 6(b).

Applying similar analysis as in section 3.2, the composition of the ZOH, the ordinary differential system modeling the water tank, and the zero-reaching event generator is a strictly causal process. The merge of discrete events is also a causal process [10]. Thus, if the switching process does not introduce any time delay, then the entire system is a feedback composition of a strictly causal process. In fact, the system exhibits Zeno behavior that the in-flow water will switch between the two water tanks infinitely many times within a finite time interval. This Zeno phenomenon will not appear if the composition is δ-causal: for example, if the switching of in-flow water from one tank to another always takes at least Δ amount of time, or if the water tank part of the system (processes within the circle in Figure 6) is implemented as a sampled differential system instead of using state-event detections.

4 Simulation Strategies

Existence and uniqueness theorems give a denotation of a system behavior. It is of practical importance to compute the behavior operationally, and to answer questions such as whether the operational semantics is a precise establishment of the denotational semantics (a.k.a. the *full abstraction* problem), and if not, how close they are.

It is not uncommon in the discussion of continuous-time and mixed-signal simulations to realize that it is impossible to represent continuous-time waveforms in digital computers and that a numerical solution of an ODE is only an approximation, and give up full abstraction immediately. Nevertheless, we believe that it is possible to develop a discrete abstraction for continuous systems and provide an *abstract operational semantics* that is compatible with the denotational semantics, and to discuss the simulation strategies in general, irrespective of the ODE solvers used.

To avoid the technicality of different kinds of ODE solvers and their numerical accuracies, we introduce a notion of ideal ODE solvers. For an ordinary differential system (1), given a known point $x(t)$ on the trajectory, a time instant $t' > t$, such that the ODE satisfies the Lipschitz condition in $[t, t']$, and known input $u[t, t')$, an *ideal ODE solver* gives the exact value of $x(t')$. So, an ideal ODE solver operates discretely. Instead of trying to represent the waveform on the entire time interval, it only computes the solution at the end point of that interval. The notion of ideal ODE solvers is not completely unrealistic. Certain kinds of ODEs can be solved analytically, such that an exact solution can be obtained on any given time instant, as long as we ignore the error of representing a real number by a floating point number, say, in double precision. A degenerate form of this concept, applying to the ODE, $\dot{x} = 1$, has been shown useful in the verification of timed automata [1]. Practical numerical ODE solvers can only give an approximation of $x(t')$, but they operate in the same discrete way as an ideal ODE solver.

Under this abstraction, the continuous-time simulation problem becomes how to find the sequence of time points, such that conditions for the uniqueness of solution are not violated in each interval. This is by no means a trivial problem, especially when the continuous dynamics interacts with discrete-event processes. The causality properties of mixed-signal processes contributes to the understanding of this abstract operational semantics through the following observations.

Observation 1 *For causal functional processes, if the input is the prefix of the potentially infinite-length input signal up to time t, then the output is the prefix of the final output signal up to at least time t.*

In most mixed-signal and hybrid system modeling environments, the processes are implemented as components with states and firings, where the state of a component at time t summaries all the inputs before time t, and the firing of a component at time t computes the new state and the output of the component at some $t' \geq t$. Causality makes "state" a well-define notion. Applying Observation 1 iteratively implies that a mixed-signal system can be simulated by computing partial behaviors chronologically, a *time-marching* strategy adopted by most mixed-signal and hybrid system simulators. That is, the simulator maintains a global, monotonically increasing notion of time, and computes the behavior of the system "step-by-step." This strategy essentially implements the constructive procedure in the Banach fixed point theorem, and will converge, in the sense of the Cantor metric, to the denotational behavior if there is one. Thus, by using an ideal ODE solver, we still obtain full abstraction.

Observation 2 *For ODS* (1) - (3) *satisfying the Lipschitz condition in* $[t, t']$, *if the values* $x(t)$ *of the state variables at time* t *are known, and the input* $u[t, t')$ *is known, then an ideal ODE solver can compute* $x(t')$.

The increase of the global notion of time from one Banach fixed point iteration to the next corresponds to the step sizes in simulations. A key issue of simulating continuous parts of a mixed-signal system is find the right t'. Under the assumption of an ideal ODE solver, for a continuous-time system, it is essential that the ODE satisfies the local Lipschitz condition in every such step. So, each simulation step size should be within the value implied by the Lipschitz condition. In numerical ODE solvers, this may be approximated by monitoring local errors or the numerical convergence of integration methods. When the system also contains discrete dynamics, discrete events may effect the local Lipschitz conditions. In practice, breakpoints can be introduced to explicitly represent the time instants when the local Lipschitz conditions are violated, and require that no ODE solving steps go across break points [16].

The operation of an ideal ODE solver also requires it to know the input $u[t, t')$ when it starts computing at time t. In the interaction of continuous and discrete dynamics, $u[t, t')$ may be generated from future discrete events, which may not be all known at time t. A practical solution for this problem is to perform optimistic execution [16], where the simulator assumes that the inputs are fully predictable and runs ahead of the global time. If the prediction is wrong, the simulator rolls back to a previous state and recomputes.

Observation 3 *For* δ*-causal processes, strictly causal processes, and indirect causal processes (i.e. causal processes with no direct I/O dependencies, for example, continuous or mixed-signal processes that have at least one differential equation type of relation from the input to the output), the output at time* t *does not directly depend on the input at* t.

This implies that we can schedule the execution order for feedback loops. That is, at time t, we can evaluate a feedback loop by first letting the δ-causal, strictly causal, and indirect causal processes to produce their outputs at t. (These outputs can be empty events.) Then, evaluate other processes in their I/O dependency order.

As a summary, even with ideal ODE solvers, a correct mixed-signal simulator still needs to support breakpoint, rollback, and proper managing of time progression. Once a full abstraction is established through ideal ODE solvers, further approximation are required when using numerical ODE solvers.

5 Conclusion

In this paper, the causality issues in mixed-signal and hybrid systems are studied as part of their denotational semantics. Using the Cantor metric, we give precise definitions of causality, strictly causality, and δ-causality in mixed-signal systems. With these definitions, we apply the Banach fixed point theorem to define the denotational behavior for these systems. These causality results validate the common mixed-signal and hybrid system simulation techniques, including the time-marching strategy, evaluation of feedback loops, step size control, and the necessity of supporting rollback.

Acknowledgments

This research is supported in part by the Ptolemy project, which is supported by DARPA/ITO, the State of California MICRO program, and the following companies: Agilent, Cadence Design Systems, Hitachi, and Philips.

References

1. Henzinger, T.A.: The theory of hybrid automata. In: Proceedings of the 11th Annual Symposium on Logic in Computer Science (LICS). IEEE Computer Society Press (1996) 278–292
2. Bornot, S., Sifakis, J.: On the composition of hybrid systems. In: Hybrid Systems: Computation and Control, HSCC'98, Berkeley, CA, April, 1998. Volume 1386 of Lecture Notes in Computer Science. Springer (1998) 49–63
3. Lygeros, J., Johansson, K.H., Simic, S.N., Zhang, J., Sastry, S.: Dynamical properties of hybrid automata. IEEE Transactions on Automatic Control **48** (2003) ,to appear.
4. van der Schaft, A., Schumacher, J.: Complementarity modeling of hybrid systems. IEEE Transactions on Automatic Control **43** (1998) 483–490
5. Lemmon, M.: On the existence of solutions to controlled hybrid automata. In: Hybrid Systems: Computation and Control, HSCC'00, Pittsburgh, PA, USA, March, 2000. Volume 1790 of Lecture Notes in Computer Science. Springer (2000) 229–242
6. Benveniste, A.: Compositional and uniform modeling of hybrid systems. IEEE Transactions on Automatic Control **43** (1998) 579–584
7. IEEE 1076.1 Working Group: VHDL 1076.1-1999: Analog and mixed-signal extensions to VHDL (1999)
8. Harman, T.L., Dabney, J.B.: Mastering Simulink 4. Prentice Hall (2001)
9. Lee, E.A., Sangiovanni-Vincentelli, A.: A framework for comparing models of computation. IEEE Transactions on CAD **17** (1998) 1217–1229
10. Lee, E.A.: Modeling concurrent real-time processes using discrete events. Annals of Software Engineering **7** (1999) 25–45
11. Royden, H.: Real Analysis. 3rd edn. Prentice Hall, Englewood Cliffs, NJ (1988)
12. Sastry, S.: Nonlinear Systems, Analysis, Stability and Control. Springer-Verlag (1999)
13. Zhang, J., Johansson, K.H., Lygeros, J., Sastry, S.: Dynamical systems revisited: Hybrid systems with Zeno executions. In Krogh, B., Lynch, N., eds.: Hybrid Systems: Computation and Control, LNCS 1790. Springer-Verlag (2000)

14. Bryant, V.: Metric Spaces. Cambridge University Press (1985)
15. Johansson, K., Egerstedt, M., Lygeros, J., Sastry, S.: On the regularization of Zeno hybrid automata. Systems and Control Letters **38** (1999) 141–150
16. Liu, J., Lee, E.A.: Component-based hierarchical modeling of systems with continuous and discrete dynamics,. In: IEEE Symposium on Computer-Aided Control System Design (CACSD'00), Anchorage, Alaska, USA. (2000) 95–100