Hybrid transition systems

Hybrid Transition Systems*

P.J.L. Cuijpers[1], M.A. Reniers[1], W.P.M.H. Heemels[2]

[1] Department of Mathematics and Computer Science
[2] Department of Electrical Engineering
Technische Universiteit Eindhoven (TU/e)
P.O. Box 513, 5600 MB Eindhoven, The Netherlands,
{P.J.L.Cuijpers, M.A.Reniers, W.P.M.H.Heemels}@tue.nl

1 Introduction

The theory of hybrid systems studies the combination of continuous and discrete behaviour. When discrete software is combined with mechanical and electrical components, or is interacting with, for example, chemical processes, an embedded system arises in which the interaction between the continuous behaviour of the components and processes and the discrete behaviour of the software is important. Although there are good methods for describing, modeling and analysing continuous behaviour (control science / system theory) as well as for analysing discrete behaviour (computer science / automata and process theory), the interaction between those two fields is largely unexplored, but received a lot of interest recently [vdSS00b, BM99, vBGR97, vdB98, BBM98]. There are only a few models that can handle (some) interaction and often these models are still dominated by one of the two original fields. Take as an example the timed process algebras (see [BM01, RGvdZvW02]) used in computer science or system theoretic methods like complementarity systems and switched systems (see [vdSS00b]). Two broader and more expressive formalisms that are in use already are hybrid automata [LSV99, LSV01] and rich time behaviour [vdSS00b]. Those are discussed in more detail furtheron in this report.

In practice, often the discrete part of a hybrid system is described and analysed using methods from computer science, while the continuous part is handled by control science. Because the analysis of the interaction between the discrete and continuous part is extremely difficult, the design of the complete system is usually such that this interaction is suppressed to a minimum. This is the main

*This work was financed by Progress/STW Grant EES5173

reason for the development of a theory on hybrid systems. If we can obtain more insight in the interaction between discrete and continuous behaviour, we can get rid of the current restrictions on the design of a hybrid system. In the remainder of this report, system theory, automata theory and process theory, are referred to as *classical* theories, as opposed to combinations of those in *hybrid* theories.

Our ultimate goal is a syntactical algebraic structure that can serve as a modeling framework for hybrid systems and in which we can do symbolic analysis. As will become clear in the next section, such an algebra should have an underlying mathematical structure that reflects the meaning of the algebraic operators. This underlying structure must be intuitive from both a control science and a computer science point of view.

In this report, we introduce hybrid transition systems as a new candidate structure to serve this goal. This model is less expressive than some of the existing hybrid formalisms, but we argue that it is more suitable as a model because, in our opinion, it gives rise to more elegant definitions of the theoretical properties that play a role in hybrid systems theory. Especially, the fact that only one mathematical structure is needed to describe discrete as well as continuous behaviour, is important when comparing hybrid transition systems to for example hybrid automata. With respect to the expressivity, we indicate which information is not modelled in the hybrid transition systems that is modelled in other formalisms. This is to ensure that no crucial information is lost due to the fact that hybrid transition systems are not as expressive.

We start with an explanation of our view on mathematical modeling and then give an overview of models (classical and hybrid) in the literature from both fields. We discuss some of their strengths and weaknesses and, after this, we come to the model of hybrid transition systems, and point out how it relates to the classical models. In order to strengthen intuition, we discuss in an informal way how several notions from the different classical theories (like *bisimulation*, *time-invariance*, *stability* and *controllability*) can be incorporated into the new context.

At some points in this report we rely on a basic knowledge of the field of topology. For an introduction into this field of mathematics, see for example [Dug66, Eis74]. A reminder of some of the basic definitions is given in the appendix, as well as an explanation of most of the notation used in this report.

2 Mathematical Modeling

In this section, we explain our view on the concept of mathematical modeling. An informal explanation is given of what a mathematical model consists of, and why. One part of mathematical modeling, the semantics, is extensively studied in this report for frameworks from computer science and system theory, as well

as for hybrid frameworks.

A mathematical formalism provides us with a structure in which we can describe systems, and in which we can analyse them. Mathematical modeling often makes use of two of such formalisms called syntax and semantics (see figure 1).

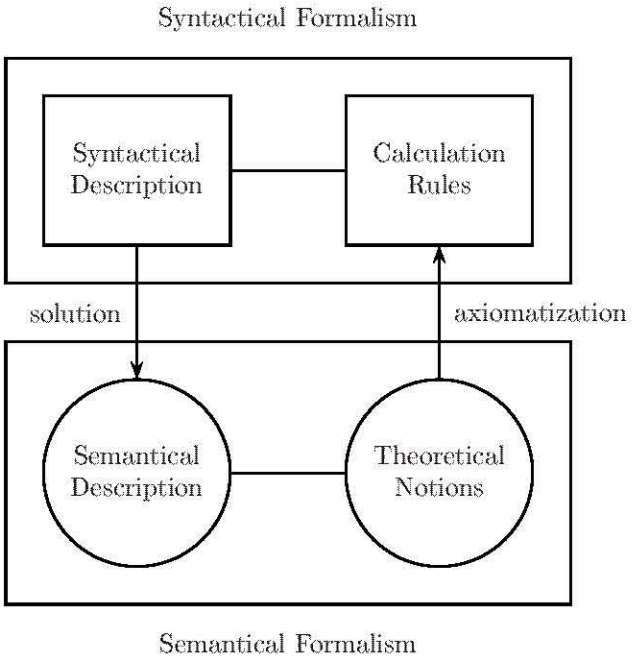

Figure 1: Mathematical Modelling

The *semantical formalism* (in short: semantics) is intended to support the modeling of a system on a low level of abstraction. The *semantical description* of a system uses a relatively simple mathematical structure. Motion, on a semantical level, could for example be modeled using functions of time to space. A computer program could be modeled using a graph-like structure called a transition system. The semantical formalism contributes to the analysis of systems by the intuitive definitions it provides of the *theoretical notions* we want to analyse. Because of its simplicity as a mathematical structure, the semantical formalism allows us to give a precise, and intuitive, definition of several notions like equivalence, stability, absence of deadlock, controllability, and observability (which terms are explained roughly throughout the text and considered in more detail in the last sections of this report).

The *syntactical formalism* (in short: syntax) is intended to facilitate a less cumbersome description of a system. In contrast to the mathematically simple description method that the semantical formalism provides, the *syntactical*

description is focussed on the ease of notation. Writing down, on paper, the complex ways in which planets move using functions of time, would be impossible because there are simply too many (infinitely many) possible evolutions, especially when no initial condition is given. Describing them using, for example, differential equations, provides us with a finite representation of the same set of functions. The high-level Pascal or C++ code of a computer program, is far more easy to write down than a transition system with the same functionality. Syntax provides a concise, finite way of handling semantical, and often infinite, mathematical objects.

This suggests that the syntactical and semantical formalism are coupled, which indeed they are. A differential equation has solutions in terms of functions of time. A piece of C++ code, although not formally (to which we return later), represents a transition system.

The contribution of syntax to the analysis of systems is through axioms and theorems that we refer to in the figure as *calculation rules*. Because syntax and semantics are coupled, the notions that are defined in the semantics, have a meaning in the syntax. The calculation rules on the syntax, should reflect these notions. Axioms (for example) usually represent notions of equivalence on the semantics, while the theorems about the stability of systems correspond to the definition of stability in semantical terms. For the analysis of systems it is important that the coupling between syntax and semantics is formal, this is one of the reasons why C++ programs are almost impossible to analyse completely. This is also one of the reasons for the development of a formal semantics for languages like UML [DMY02, GPP98, EBF+98], and χ [BK02, Are96, vBR00] that were originally intended for other purposes, like simulation. Typical syntactical languages that were developed with the intention of analysis from the beginning, are process algebras like ACP (Algebra of Communicating Processes) [BW90, Fok98], μCRL (micro Common Representation Language) [GR01, RGvdZvW02] and CCS (Calculus of Communicating Systems) [Mil80].

In figure 2, a graphical representation is given of the general aim of our efforts. The figure shows that we want to combine the syntax used by system theorists and the syntax used by process theorists into a new hybrid syntax. A similar integration is aimed at for the semantics of both fields. It is important for the user of the hybrid theory that a classical syntactical statement has the same meaning in the hybrid semantics as it did in the classical semantics (after the necessary translations of course). In more technical terms, the figure must be commuting.

In this report, we concentrate on the semantical part of the theory. A good semantics helps in finding a good syntax and makes the formalisation of intuitions possible. In sections 3 and 4, we take a better look at classical semantics. What models are used and what notions are of interest on those models? After that we study possible combinations of models in three different hybrid semantics in section 5. Two of those semantics are taken from literature, one is newly intro-

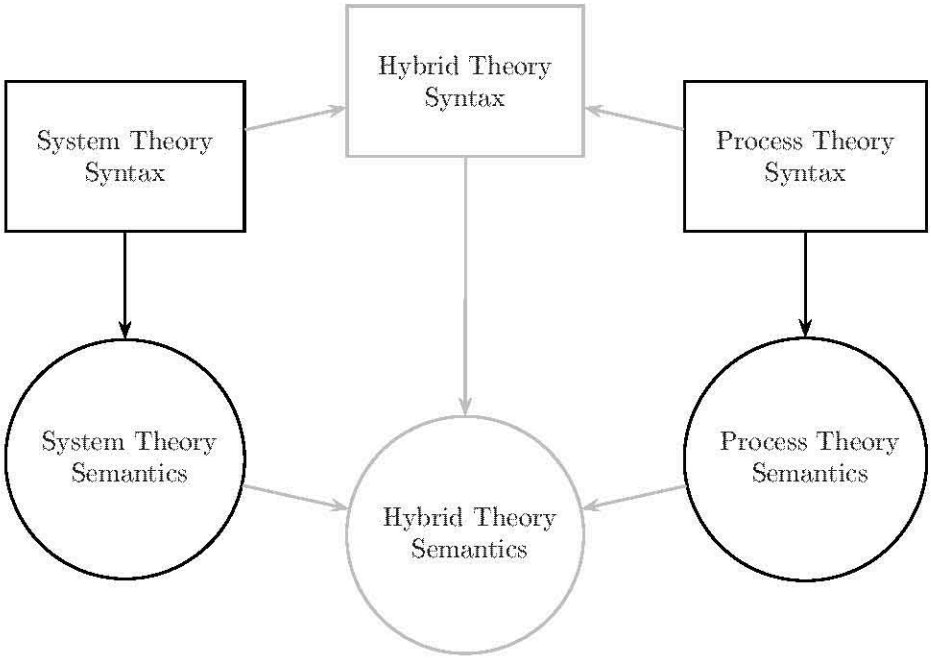

Figure 2: Developing Hybrid Theory

duced. We compare the three different hybrid models and discuss our reasons to choose hybrid transition systems. The fact that we do not plan to consider a particular hybrid syntax yet, causes the problem that we cannot close the commuting diagram completely. Nevertheless, if we can show that models in the classical semantics can be expressed in terms of the new hybrid semantics without loss of crucial information, we know that the new semantics is at least strong enough to contain the classical formalisms. In section 6, we provide the necessary proofs to show that the chosen semantics is indeed expressive enough for our needs. Finally, we spend three sections, 7, 8 and 9, on an overview of frequently used notions from classical theory in terms of hybrid transition systems. Furthermore, we give an assessment of the specific hybrid problems that arise when defining those notions.

3 A General Model of Systems

In this section, we introduce a general form for defining semantical models for systems. Furthermore, we study the semantical formalisms of the classical theories, system theory and process theory, in this form. In later sections,

the general form makes it easy to combine the classical semantical models into hybrid semantical models.

The semantics of classical theories all give their view of the world through the definition of a certain kind of system. Transition systems are a semantical formalism that has been adopted by most computer scientists (amongst many others see: [GR01, BW90, Fok98, BK02]). Behavioural systems [PW98] and Sontag machines [Son98, Phi01] are two semantical formalisms in use in control science. Behavioural system semantics is focussed on giving definitions without referring to input/output considerations as much as possible, while Sontag machines take a more operational view in modeling systems, that is close to the transition system formalism. We have attempted to give a unified view on systems that is more or less in line with the literature available on all the different classical semantics we study in this report.

A system is a phenomenon of memory, interaction and time.

This leads to the following partial definition, in which only the concept of mathematical structure is still open. Specific instantiations of this mathematical structure are be used to complete the definition for the different semantical models introduced lateron.

Definition 1 (System) *A system is a tuple $\langle X, \Sigma, T, \phi \rangle$, in which X denotes the state or memory*[1] *of the system, Σ denotes the interaction space (also called signal space, control space, or alphabet), and T denotes the time axis. The type of the system is determined by the structure ϕ, a mathematical structure on the state and interaction spaces and the time axis of the system.*

The above definition states that a system consists of the spaces X,Σ and T, and a structure ϕ on those spaces. As we mentioned before, the types of systems that are most important for our goal can be found in the literature on classical theories, although in many books on control science in particular, the semantical formalism is not formally defined. We believe the following systems form good representatives of the semantics used throughout literature.

For all systems, we assume there is a total ordering on the time axis T. One of the reasons for this is the common use of intervals on T, which are only defined if T is totally ordered (see appendix A). Another, more important reason, is that a total ordering supports notions like "past" and "future" of a system evolution. This is discussed further on in this section.

- *Timed Labeled Transition Systems*, in which $\phi \subseteq (X \times T) \times \Sigma \times (X \times T)$ is a relation that models how a state $x \in X$ at time $t \in T$ can evolve into

[1] The words "state" and "memory" are more or less synonymous, and used as such throughout this report.

another state $x' \in X$ at time $t' \in T$ due to an interaction $\sigma \in \Sigma$. Usually, $(x, t, \sigma, x', t') \in \phi$ is denoted $\langle x, t \rangle \xrightarrow{\sigma} \langle x', t' \rangle$. Timed labeled transition systems, in this particular or a similar form, are a common way of modeling systems in computer science [BM02, BM01, AD94, GP95, BK02]. An impression of a timed labeled transition system is given in figure 3.

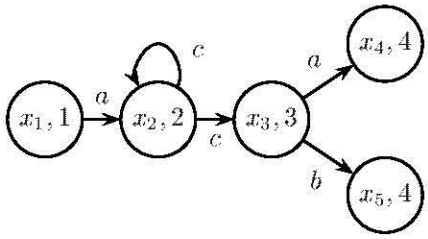

Figure 3: Example of a Timed Labelled Transition System, a transition relation between states $x_1, x_2, x_3, x_4, x_5 \in X$, at times $1, 2, 3, 4 \in T$, labelled with interactions $a, b, c \in \Sigma$

- *Behavioural Systems*, in which $\phi \subseteq T \mapsto (X \times \Sigma)$ is a set of functions modeling the possible evolutions of state and interaction of a system through time. Behavioural systems form a particularly intuitive semantics to specify the solutions of, for example, differential equations, and are used in control science mainly by those who seek for a meta-theoretic approach to control [PW98, Wei91]. Although the structure allows for arbitrary partial functions $p \in \phi$, usually the domain of p is assumed to be an interval. Usually, the (mathematically dubious) notation (x, σ) is used instead of p, in which $x \in T \mapsto X$ and $\sigma \in T \mapsto \Sigma$ denote functions representing the state and signal trajectories, such that for all $t \in Dom(p)$ we have $p(t) = (x(t), \sigma(t))$ and $Dom(x) = Dom(\sigma) = Dom(p)$. A drawn impression of a behavioural system is given in figure 4.

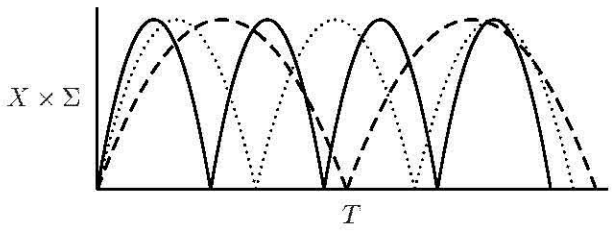

Figure 4: Example of a Behavioural System, a system consisting of a set of functions from time to interactions and states

- *Sontag Machines*, in which $\phi \in (X \times (T \mapsto \Sigma)) \mapsto X$ is a different kind of (functional) transition relation with a labeling consisting of partial functions of time to interaction. The relation reflects in an operational way how certain partial signal trajectories let the system evolve from one state to another. A drawback of Sontag machines is that they do not support non-determinism. The transition relation is functional, and hence every pair of state $x \in X$ and interaction function $\sigma \in T \mapsto \Sigma$ evolves into at most one subsequent state $x' = \phi(x, \sigma)$. Due to this, Sontag machines can only be used as solutions of differential equations with unique solutions for a given interaction signal and initial state [Son98]. The labels of the transitions in Sontag machines are partial functions that have an interval domain. More specifically, in Sontag machines we have for all interaction functions σ that $Dom(\sigma) = [t..t')$ for some $t, t' \in T$. The domains are so called *left-closed right-open intervals*. This is done to make sure that subsequent transitions do not overlap each other. Hence, the total signal trajectory as used in the behavioural context can be recovered from concatenation of the transitions. There are some more constraints on Sontag machines that, for example, guarantee that also the state trajectory can be recovered. One of them is that from each state x there is at least one outgoing transition; i.e. there exists a σ and x' such that $x' = \phi(x, \sigma)$. In computerscience terms this is called *absence of deadlock*. Because the other constraints are not important for the definitions in this report, we d not go into detail about them here. A drawn impression of a Sontag machine is given in figure 5.

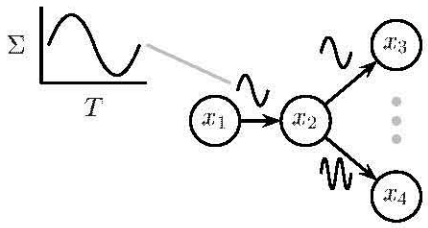

Figure 5: Example of a Sontag Machine, a functional transition relation on states $x_1, x_2, x_3, x_4 \in X$, labelled with partial functions from time to interactions space

So far, we only addressed the *semantics* of classical theories, and also in the remainder of this report we restrict ourselves to semantical issues. However, in order to fill in figure 1 for the classical theories, we have to at least mention an example of syntax here. For computer science, we use as an example the process algebraic description method, while control science is associated with descriptions using differential equations. Furthermore, as an example of calculation rules, consider the process algebra axioms with respect to bisimulation

equivalence [GR01, Fok98, BW90], and realise that Lyapunov developed useful calculation rules to establish the stability of systems (see for example [TSH01]). In figure 6, we have depicted these examples once more. Note that we only depicted the behavioural systems case of control science. The case for Sontag machines is, except of course for the semantics, exactly the same.

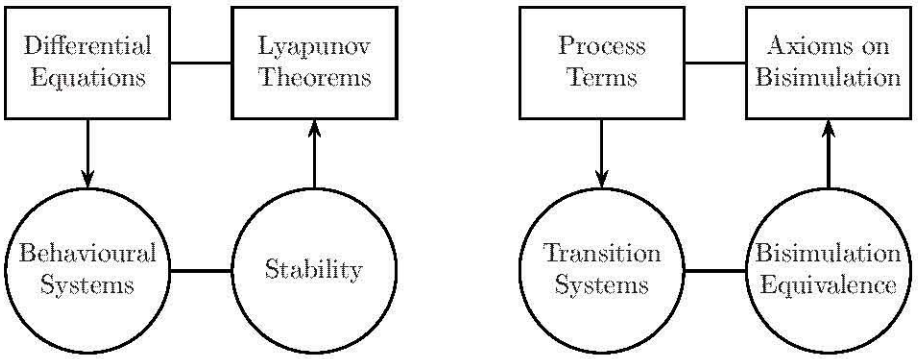

Figure 6: Examples in the Mathematical Modelling Scheme of Classical Theories

In the previous section, we mentioned that the commuting figure 2, cannot be closed completely because we do not have a hybrid syntax yet. Furtheron, we make some assumptions on the kind of syntax that is involved when we show that hybrid transition systems are expressive enough for our needs. Before we get into this, let us concentrate on the theoretical notions that play a role in classical theory.

4 Theoretical Notions on Systems

From the description of the different systems in the previous section, one may already have guessed that an ever important notion on systems is that of *evolution*. Transition systems and Sontag machines describe the evolution of one state of the system into another, while behavioural systems describe the complete evolution of a system over time. We do not attempt here to give a general formalisation of evolution, because this would lead us away too much from our main goal: a formalism for hybrid systems. The notion of evolution, however, is important because it allows us to specify certain intuitions we have about time and state.

- Given the state of a system at a certain time, the future evolution of a system is independent of the past.
- During an evolution time may not run backwards.

In this section, we investigate the intuition behind the notion of evolution and other notions that are important in general for the theory of any kind of system.

The intuition that evolutions of a system are independent of the past, given a certain state and time, can be easily illustrated using behavioural systems. In behavioural systems, the evolutions are simply the elements of ϕ. To support the intuition, behavioural systems are required to have the property of state. This property was introduced in [PW98], and describes how the concatenation of evolutions, at a certain state and time, forms new evolutions.

Definition 2 (Property of State) *A behavioural system* $\langle X, \Sigma, T, \phi \rangle$ *has the* property of state *if for all evolutions* $(x, \sigma), (x', \sigma') \in \phi$ *and times* $t \in T$ *we find*

$$x(t) = x'(t) \;\Rightarrow\; (x \ominus_t x', \sigma \ominus_t \sigma') \in \phi \;,$$

in which

$$(f \ominus_t g)(\tau) = \begin{cases} f(\tau) & \tau < t, \\ g(\tau) & \tau \geq t. \end{cases}$$

In [PW98], it is shown that the behavioural systems that arise as solutions of differential equations and other system theoretic descriptions indeed have the property of state.

The other intuition we have on evolutions, namely that time does not run backwards, is best illustrated using the evolutions from computer science, called runs.

Definition 3 (Run) *A* run *of a timed labeled transition system* $\langle X, \Sigma, T, \phi \rangle$, *is a pair* (x, τ, σ) *of a state trajectory* $x \in \mathbb{N} \mapsto X$, *a time trajectory* $\tau \in \mathbb{N} \mapsto T$, *and an interaction trajectory* $\sigma \in \mathbb{N} \mapsto \Sigma$ *such that*

- $Dom(x)$, $Dom(\tau)$, *and* $Dom(\sigma)$ *are intervals in* $\mathbb{N}$;
- $Dom(\tau) = Dom(x)$, *and* $0 \in Dom(x)$;
- $Dom(\sigma) \subseteq Dom(x) \;\wedge\; \forall_{n \in Dom(x)}\; n \in Dom(\sigma) \;\Rightarrow\; n+1 \in Dom(x)$;
- $\forall_{n \in Dom(\sigma)}\; \langle x(n), \tau(n) \rangle \overset{\sigma(n)}{\rightarrow} \langle x(n+1), \tau(n+1) \rangle$;

We define the length of a run to be the cardinality of $Dom(\sigma)$.

The intuition that time does not run backwards is modeled by an extra constraint: $\forall_{n \in Dom(\sigma)}\; \tau(n) \leq \tau(n+1)$ (note that multiple actions may take place after each other at the same time). From now on, we only consider the class of

timed transition systems that satisfy this constraint, i.e. those timed transition systems such that $\langle x, t \rangle \xrightarrow{a} \langle x', t' \rangle$ (with $x, x' \in X$, $a \in \Sigma$ and $t, t' \in T$) implies $t \leq t'$. For runs, the first intuition that future evolutions are independent of the past, given a state and a time, is automatic.

Other properties that are based on the evolution of systems are notions from control science like

- observability: one can know the state of the system by only observing the evolution of the interaction values;
- controllability: one can steer the evolution of a system by forcing certain interaction values;
- stability: the evolutions of a system are all bounded;

and notions from computer science like

- deadlock: the system can evolve into a state from which there are no future evolutions;
- bisimulation equivalence: two systems cannot be distinguished from observing and manipulating the evolution of interaction values only.

Usually, part of the analysis of systems is focussed on assessing whether these properties hold or not. In section 9, we argue that these notions automatically obtain meaning in the hybrid semantics as soon as we have a proper notion of evolution for that semantics. In the next section, we combine the classical semantics we studied so far, into several examples of hybrid semantics.

5 Hybrid Systems

Recall that we have three classical semantical formalisms that we would like to incorporate into one hybrid formalism in such a way that at least the syntactical formalisms associated with them are supported. In the case of computer science we would like to support a process algebra kind of syntax, in the case of control science we would like to support differential equations. Loosely speaking, the three semantical formalisms are the following:

- Timed Labeled Transition Systems: $\phi \subseteq (X \times T) \times \Sigma \times (X \times T)$;
- Behavioural Systems: $\phi \subseteq T \mapsto (X \times \Sigma)$;
- Sontag Machines: $\phi \in (X \times (T \mapsto \Sigma)) \mapsto X$.

Actually, Sontag machines are in itself not even sufficient to support differential equations, because differential equations with multiple solutions for the same initial condition cannot be modeled. Nevertheless, enhancing these machines slightly to incorporate non-determinism might solve this problem. By mixing the definition of (non-deterministic) Sontag machines with timed transition systems we obtain the following definition. Note, that we divide the signal space into a continuous and discrete part for clarity of the definition only. It has no formal consequences since the two parts need not be disjunct, but it clearly shows which part of the definition originates from computer science, and which part originates from system theory. Furthermore, we use closed intevals for labelling instead of left-closed right-open intervals as was the case with Sontag machines. This proves usefull lateron, in the proofs of expressivity of hybrid transition systems. Although we suspect that these proofs can still be given when left-closed right-open intervals are used, it would unnecessarily complicate them.

Definition 4 (Hybrid Transition System) *A* hybrid transition system *is a tuple* $\langle X, \Sigma, T, \phi \rangle$ *with* T *totally ordered by* $\leq$ *and a signal space* $\Sigma = \Sigma_C \cup \Sigma_D$ *divided in a continuous and discrete part. Furthermore, it has a hybrid transition relation*

$$\phi \subseteq (X \times T) \times ((T \mapsto \Sigma_C) \cup \Sigma_D) \times (X \times T) \ .$$

We use $\langle x, t \rangle \xrightarrow{\sigma} \langle x', t' \rangle$ *to denote* $(x, t, \sigma, x', t') \in \phi$*, and we demand that the labels only carry information about the duration of a transition, not about the precise timing. This means we restrict ourselves to partial functions* $\sigma \in T \mapsto \Sigma_C$ *that have a closed interval domain of the form* $Dom(\sigma) = [0..t' - t]$.

Note that it is possible to write down this definition a little more concise, since the signals from Σ_D can also be regarded as continuous signals on a one element domain, the interval $[0..0]$. As with timed transition systems, we usually assume that every transition $\langle x, t \rangle \xrightarrow{\sigma} \langle x', t' \rangle$ is time increasing such that $t \leq t'$. A graphical impression of a hybrid transition system is depicted in figure 7.

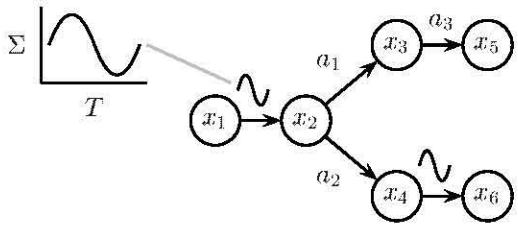

Figure 7: Example of a Hybrid Transition System

From literature, we have two other possible semantics for hybrid systems that merge labeled transition systems with behavioural systems. The first one, hybrid automata (as defined in [LSV99, LSV01]), takes the union of the complete structures. Usually there are a few constraints (like the property of state) on the behavioural part of ϕ, but we do not elaborate on those technicalities here.

Definition 5 (Hybrid Automaton) *A* hybrid automaton *consists of a tuple* $\langle X, \Sigma, T, \phi \rangle$ *with* T *totally ordered by* $\leq$*, a signal space* $\Sigma = \Sigma_C \cup \Sigma_D$ *divided in a continuous and discrete part, and a structure*

$$\phi \subseteq ((X \times T) \times \Sigma_D \times (X \times T)) \cup (T \mapsto (X \times \Sigma_C)) \ ,$$

in which the partial functions $(x, \sigma) \in \phi$ *are again assumed to have an (arbitrary) interval domain.*

A graphical impression of this is difficult because two different mathematical structures are used in ϕ.

The second alternative for hybrid transition systems is hybrid behaviours. They are behavioural systems that extend the time axis to be able to support multiple transitions at a single time instance, as is the case with the runs defined in section 3. In [vdSS00b, vdSS00a], the notion of *time enrichment* was introduced to this extend. Here we use a slightly different definition that has the same power as the time enrichment, based on a Cartesian product of time and ordinal numbers (see for example [Kun88]). We denote the collection of ordinal numbers as Ω.

Definition 6 (Hybrid Behavioural System) *A* hybrid behavioural system *consists of a tuple* $\langle X, \Sigma, T, \phi \rangle$ *with* T *totally ordered*[2] *by* $\leq$*, the signal space* $\Sigma = \Sigma_C \cup \Sigma_D$ *divided into a continuous and a discrete part, and having the extended behavioural structure*

$$\phi \subseteq (T \times \Omega) \mapsto (X \times (\Sigma_C \cup \Sigma_D)) \ .$$

The set $(T \times \Omega)$ *is totally ordered by the relation* $\preceq$ *such that*

$$(t, n) \preceq (t', n') \ \Leftrightarrow \ (t < t') \ \vee \ (t = t' \ \wedge \ n \leq n').$$

As before, the behavioural structure is assumed to have the property of state. According to [vdSS00b], the domain of $(x, \sigma) \in \phi$ *is special. It is an interval with respect to* T*, and locally with respect to* Ω*, but it is not an interval with respect to* $\preceq$*.*

[2]The natural ordering on the ordinal numbers Ω is also denoted by $\leq$, but since the type will always be clear from the context, no confusion should arise.

- *abstract interval:*
  $\forall_{t,t',t''\in T}\forall_{n,n''\in\Omega}\, t \leq t' \leq t'' \wedge (t,n),\, (t'',n'') \in Dom(x) \;\Rightarrow\; \exists_{n'\in\Omega}\, (t',n') \in Dom(x)$;

- *local closed interval:*
  $\forall_{n,n',n''\in\Omega}\forall_{t\in T}\; n \leq n' \leq n'' \;\wedge\; (t,n),\, (t,n'') \in Dom(x) \;\Rightarrow\; (t,n') \in Dom(x)$.

The most important difference between this definition and the definition of (normal) behavioural systems is that the time-elements have a successor. The successor of (t,n) simply is $(t,n+1)$. This means that one can speak of a sequence of multiple actions that occur at the same time $t \in T$. The special domain restrictions then indicate that the domain of the evolutions is an interval with respect to T, but that those sequences of actions do not have to be of length Ω. A graphical representation of a hybrid behavioural system is depicted in figure 8. In this picture, the arrows depict discrete signals, while the arcs depict continuous signal evolutions.

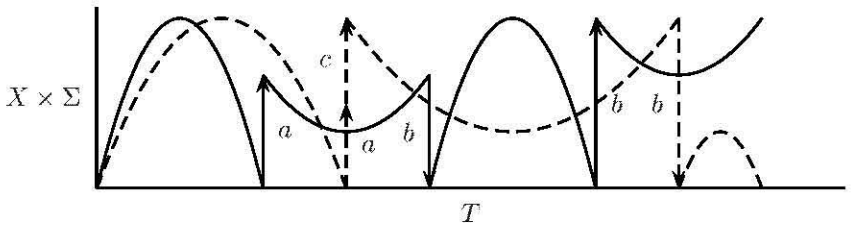

Figure 8: Example of a Hybrid Behavioural System

Note that actions in this model are all instantaneous. The actions $\langle x,t\rangle \xrightarrow{\sigma} \langle x',t'\rangle$ from a timed transition system in which $t \neq t'$ are interpreted as an instantaneous action at time t followed by a delay until time t'. This is a common viewpoint in hybrid systems theory.

Since these three definitions of different hybrid semantical models are all expressive enough for our needs (which we establish formally in the next section for the one we choose), choosing one cannot be done on the basis of technical mathematical arguments. It merely becomes a matter of taste. To our feeling, in hybrid automata the different classical theories are still too much separated. We feel that the union of two rather different structures gives a new structure that is not only slightly counterintuitive, but also mathematically awkward. As we mentioned in section 2, a semantical formalism should have a *simple* mathematical structure. To a certain extend, hybrid transition systems possess this same awkward union in the labeling of transitions. However, the model seems to evolve naturally from both the timed transition system point of view and the

Sontag machine point of view. The operational view supported by hybrid transition systems provides us with a strong intuition about the model that complies with the intuitions on both classical theories. Furthermore, in computer science there is already some experience with transition systems with two types of labels in the context of timing (for example [BM02, BB91, Hen96, BK02, BM01]). The third model, hybrid behavioural systems, is oriented on the complete evolutions of a system. This view leads to a focus on trace equivalence, rather than on bisimulation equivalence[3]. Since our focus is on the latter, stronger notion, we decided that the hybrid transition system model forms the most suitable semantics for the development of a hybrid syntactical algebraic framework. It has a clear operational and state-oriented view, and in our opinion is a more elegant structure than hybrid automata. Computer science notions like bisimulation equivalence on states can be defined in a natural way and will help us to abstract away from the precise contents of the states. The topological influence from control science will provide us with a notion of continuity and aid us to some extend in handling the deviations in the precise value of the state that occur in physical systems. However, we realise that also the view on complete evolutions is valuable, especially when considering for example notions like stability, where the boundedness of the evolution cannot be simply reduced to the boundedness of single transitions. It will always be possible, of course, to define such a notion of evolution on hybrid transition systems.

6 Translations

From section 2, we know that, in order to be sure that a proposed hybrid semantics is indeed a suitable semantics for describing hybrid systems, we have to translate the classical semantics into the hybrid one. In this section, we do that for the chosen semantics of hybrid transition systems.

The case of timed labeled transition systems is trivial. A timed labeled transition system *is* a hybrid transition system.

In the case of Sontag machines the translation is a little more difficult due to differences in the handeling of the domain of the signals σ. The first difference is, that Sontag machines are labelled with functions on a left-closed right-open domain, while hybrid transition systems have labels with completely closed intervals as domain. The translation from Sontag machines to hybrid transition systems is possible if we can find a "last point" to close the domain of the Sontag transition. Suppose we have a transition from (x, t) into (x', t') using some label σ defined on a domain $[t..t')$. The property of Sontag machines that there is absence of deadlock (see also [Son98] and various lemmas on the existence of

[3]Note that the expressivity argument implies that defining bisimulation on behavioural systems must be possible. However, these definitions become cumbersome due to the evolution oriented structure.

solutions of differential and other behavioural equations), gives us the opportunity to create a non-empty set of "first points" of transitions starting in (x', t'). Any of those points will do to define $\sigma(t')$.

The second difference between Sontag machines and hybrid transition systems is, that there is information on the timing in the labelling of Sontag machines. In our opinion, this is not a good choice because, as we see furtheron, the labeling information shows what is visible to an external observer. Our intuition (which is perhaps a little philosophical in nature) is that duration is visible to an observer, but the exact time of a system is not. Duration is the only quantity regarding time that can actually be measured. Therefore, every σ label on a domain $[t..t')$ is translated into a label on the domain $[0..t'-t]$. Apart from this difference, every Sontag machine is a hybrid transition system.

The case of behavioural systems deserves more attention, because without further restriction, behavioural systems cannot be translated into hybrid transition systems without loss of information! Nevertheless, we are able to give necessary and sufficient conditions for the translatability, from which it becomes more clear which information exactly is lost. Furthermore, we argue that this information is not considered to be of importance thus far in system theory or computer science, nor in the field of hybrid systems research. Therefore, at least for the time being, hybrid transition systems are expressive enough for our needs.

The translation we use to turn a behaviour into a hybrid transition system is the following:

$$\begin{array}{|l} \text{if } (x,\sigma) \in \phi \text{ and } t, t' \in \mathit{Dom}(x) \text{ and } t \leq t' \\ \text{then } \langle\, x(t), t \,\rangle \overset{(\sigma|_{[t..t')})^{-t}}{\rightarrow} \langle\, x(t'), t' \,\rangle. \end{array}$$

In which we use $\sigma|_D$ to denote the restriction of the function σ to the subdomain $D \subseteq \mathit{Dom}(\sigma)$ and σ^t to denote the shifted function σ such that $\sigma^t(t'+t) = \sigma(t')$ for all $t' \in \mathit{Dom}(\sigma)$ and undefined elsewhere.

In order to prove that no information is lost, we use the translation back that states:

$$\begin{array}{|l} \text{if } x \text{ continuous, and for all } t, t' \in \mathit{Dom}(x) \text{ such that } t \leq t' \\ \text{we find} \langle\, x(t), t \,\rangle \overset{(\sigma|_{[t..t')})^{-t}}{\rightarrow} \langle\, x(t'), t' \,\rangle, \text{then } (x,\sigma) \in \phi'. \end{array}$$

In the remainder of this section, we focus on proving that $\phi = \phi'$ for a certain class of behaviours ϕ, meaning that behavioural systems that fit in this class can be successfully translated. However, let us first look at an example of a behaviour that cannot be translated in this way.

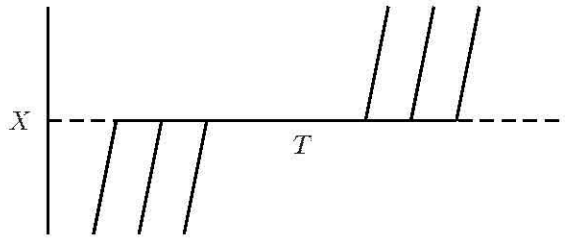

Figure 9: Example of a behavioural system that cannot be translated

Consider the behaviour depicted in figure 9. The only relevant behaviour of this system is the state behaviour, which consists of many "instable" evolutions, that "start" in minus infinity and "end" in plus infinity. Stated this way, the 0-function is clearly not part of this behaviour.

Now, if we translate this behaviour into a hybrid transition system, by constructing transitions between two states-time pairs if they are connected by a partial evolution, this information about the 0-function is lost. Obviously, the states $(0, t)$ and $(0, t')$ are connected by some evolution for all $t \leq t'$. Therefore, there is no way to decide from the transitions only, that the 0-function is not in the behaviour of the system.

Next, we show that if the behaviour has the property that it is *two-point refutable* in its evolutions, meaning that every function that is not an evolution of the system, can be recognized on the basis of two points in time, then it can be translated without loss of information.

Definition 7 (Two-point refutability) *A behavioural system* $\langle X, \Sigma, T, \phi \rangle$ *is* two-point refutable, *if for any* $(x, \sigma) \in T \mapsto (X \times \Sigma)$ *such that* $(x, \sigma) \notin \phi$ *there exist two points* $t, t' \in Dom(x)$ *with* $t < t'$ *such that for every* $(x', \sigma') \in \phi$ *either* $x(t) \neq x'(t)$, $x(t') \neq x'(t')$ *or* $\sigma|_{[t..t']} \neq \sigma'|_{[t..t']}$.

Theorem 1 *A behavioural system* $\langle X, \Sigma, T, \phi \rangle$ *can be translated into a hybrid transition system without loss of information, if it is two-point refutable in the state evolutions.*

Proof Using the translation and reverse translation above, it is trivial to see that $\phi \subseteq \phi'$, therefore we focus on the other case, to prove that $\phi' \subseteq \phi$ and assume: $(x, \sigma) \in \phi'$.
Using the fact that there is only one rule in the reverse translation, we conclude:
$\forall_{t,t' \in Dom(x)} \langle x(t), t \rangle \overset{(\sigma|_{[t..t']})^{-t}}{\rightarrow} \langle x(t'), t' \rangle$,
and using the translation from behaviours to hybrid transition systems we get:

$$\forall_{t,t' \in Dom(x)} \exists_{(x',\sigma') \in \phi} \begin{cases} x(t) = x'(t) & \wedge \\ x(t') = x'(t') & \wedge \\ \sigma|_{[t..t']} = \sigma'|_{[t..t']} & . \end{cases}$$

Using DeMorgan, we obtain:

$$\neg\exists_{t,t' \in Dom(x)} \forall_{(x',\sigma') \in \phi} \begin{cases} x(t) \neq x'(t) & \vee \\ x(t') \neq x'(t') & \vee \\ \sigma|_{[t..t']} \neq \sigma'|_{[t..t']} & . \end{cases}$$

And using modus tollens on the property of two point refutability we find:
$(x, \sigma) \in \phi$,
which concludes the proof. ⊠

Moreover, for the translations given above, two-point refutability of the state is even a necessary condition to avoid information loss.

Theorem 2 *A behavioural system $\langle X, \Sigma, T, \phi \rangle$ can only be translated into a hybrid transition system without loss of information using the translations above, if it is two-point refutable in the state evolutions.*

Proof We proof this fact, by observing that the set ϕ' has the property of two-point refutability of the state evolutions. This follows immediately out of its construction from an arbitrary hybrid transition system. If a certain function (x, σ) is not in ϕ', then there are at least two points $t, t' \in Dom(x)$ such that the transition $\langle x(t) \rangle \overset{\sigma|_{[t..t']}}{\rightarrow} \langle x(t') \rangle$ is *not* in the hybrid transition system. The absence of that transition implies that none of the functions in ϕ' are equal to the one under study at these two points. Therefore ϕ' is two-point refutable. In conlcusion, since the result of the reverse translation is a two-point refutable system, the original system must have this property too in order to obtain $\phi = \phi'$. ⊠

Now, given the property of state on a behaviour, we can loosen the constraint of two-point refutability a bit. We can show that *finite set refutability* then is sufficient (and by definition necessary) to imply two-point refutability.

Definition 8 (Finite set refutability) *A behavioural system $\langle X, \Sigma, T, \phi \rangle$ is* two-point refutable, *if for any $(x, \sigma) \in T \mapsto (X \times \Sigma)$ such that $(x, \sigma) \notin \phi$ there exists a finite set $D \subseteq [t..t'] \subseteq Dom(x)$ such that for every $(x', \sigma') \in \phi$ either $x|_D \neq x'|_D$ or $\sigma|_{[t..t']} \neq \sigma'|_{[t..t']}$.*

Note that, as was the case with two-point refutability, the signals σ and σ' in the definition are still compared over the whole interval because they are completely visible on the transitions. Hence it is not necessary to restrict the comparison to a finite set of points.

Theorem 3 *A behavioural system* $\langle X, \Sigma, T, \phi \rangle$ *that has property of state and is finite set refutable, is also two-point refutable in its state evolutions.*

Proof The proof of this is by induction on the size of D. We derive a contradiction out of the statement that D with $|D| > 2$ has no strict subset on the basis of which it is still refutable.

Suppose, for some $(x, \sigma) \notin \phi$ that

$$\exists_{D \subseteq [t..t'] \subseteq Dom(x), |D|=N} \forall_{(x',\sigma') \in \phi} \left\{ \begin{array}{ll} x|_D \neq x'|_D & \vee \\ \sigma_{[t..t']} \neq \sigma'|_{[t..t']} & , \end{array} \right.$$

and furthermore

$$\forall_{D' \subset D} \exists_{(x',\sigma') \in \phi} \left\{ \begin{array}{ll} x|_D = x'|_D & \wedge \\ \sigma_{[t..t']} = \sigma'|_{[t..t']} & . \end{array} \right.$$

with $|D| > 2$. From this last assumption we conclude that for every two points $a, b \in D$, forming a strict subset of D, there is a trajectory that connects the two.

$$\forall_{a,b \in D} \exists_{(x',\sigma') \in \phi} \left\{ \begin{array}{ll} x(a) = x'(a) & \wedge \\ x(b) = x'(b) & \wedge \\ \sigma_{[t..t']} = \sigma'|_{[t..t']} & . \end{array} \right.$$

We order the elements in D using an order preserving map $\tau \in \mathbb{N} \to D$ and conclude

$$\forall_{n<N} \exists_{(x',\sigma') \in \phi} \left\{ \begin{array}{ll} x(\tau(n)) = x'(\tau(n)) & \wedge \\ x(\tau(n+1)) = x'(\tau(n+1)) & \wedge \\ \sigma_{[t..t']} = \sigma'|_{[t..t']} & . \end{array} \right.$$

By applying this twice, we get

$$\forall_{n<N-1} \exists_{(x',\sigma'),(x'',\sigma'') \in \phi} \left\{ \begin{array}{ll} x(\tau(n)) = x'(\tau(n)) & \wedge \\ x(\tau(n+1)) = x'(\tau(n+1)) = x''(\tau(n+2)) & \wedge \\ x'(\tau(n+2)) = x''(\tau(n+2)) & \wedge \\ \sigma_{[t..t']} = \sigma'|_{[t..t']} & . \end{array} \right.$$

And then, by property of state, we may concatenate (x', σ') and (x'', σ'') to conclude

$$\forall_{n<N-1} \exists_{(x',\sigma') \in \phi} \left\{ \begin{array}{ll} x(\tau(n)) = x'(\tau(n)) & \wedge \\ x(\tau(n+1)) = x'(\tau(n+1)) = x''(\tau(n+2)) & \wedge \\ x'(\tau(n+2)) = x''(\tau(n+2)) & \wedge \\ \sigma_{[t..t']} = \sigma'|_{[t..t']} & . \end{array} \right.$$

With induction we then may conclude

$$\exists_{(x',\sigma') \in \phi} \forall_{n<N} \left\{ \begin{array}{ll} x(\tau(n)) = x'(\tau(n)) & \wedge \\ \sigma_{[t..t']} = \sigma'|_{[t..t']} & . \end{array} \right.$$

From which we derive

$$\exists_{(x',\sigma') \in \phi} \forall_{d \in D} \left\{ \begin{array}{ll} x(d) = x'(d) & \wedge \\ \sigma_{[t..t']} = \sigma'|_{[t..t']} & . \end{array} \right.$$

And using different notation

$$\exists_{(x',\sigma') \in \phi} \left\{ \begin{array}{ll} x|_D = x'|_D & \wedge \\ \sigma_{[t..t']} = \sigma'|_{[t..t']} & . \end{array} \right.$$

Which is in contradiction with the fact that (x, σ) is refutable on the basis of D. Hence, if a behavioural system is finite set refutable and has property of state, then it is two-point refutable, which concludes the proof. ⊠

Intuitively, finite set refutability means that if a certain function can be intersected arbitrarily often by piecewise concatenations of evolutions, then this function is an evolution in itself. Note that this does not necessarily mean that the function can be approximated by concatenation of evolutions. Since there is no notion of measure defined on evolutions here, approximation is not an issue. An example of a behaviour that is not finite set refutable, is the one depicted in figure 10. The set of all possible sine waves, when closed under property of state, still does not contain the 0-function, although it can be intersected arbitrarily often, by concatenation of sine waves that cross the axis.

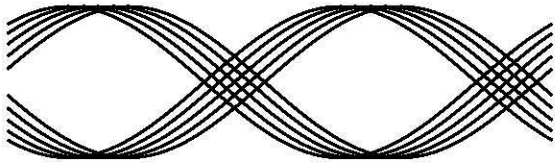

Figure 10: Impression of a non finite-set refutable set of evolutions

Looking at it from a different perspective, we see that every crossing of evolutions can be regarded as a branching of the transition system. This means that if we translate a behavioural system without finite set refutability into a hybrid transition system, then we loose information about the continuous branching options of that system. We cannot conclude the difference between a system that has a certain branching at arbitrarily small finite times apart, and a system that branches continuously.

This all, makes clear that hybrid transition systems are in a sense less expressive than behavioural systems. Whether this loss in expressivity leads to problems is an entirely different matter. There are two reasons for assuming that it does not.

The first reason, is that the hybrid transition systems are intended as an underlying model for a syntactical formalism, and the systems that are described currently in the separate worlds of computer science and system theory do not consider continuous branching. More importantly, the topic of continuous branching has not (as far as we are aware) played a role in hybrid systems research yet. Therefore, we expect that the different system compositions that we are going to study in our model for hybrid systems, do not make use of the difference between continuous branching and arbitrarily short finite-time branching. Thus, the difference between finite set refutable behaviours and behaviours that do not have this property will not be visible through any of those compositions.

The second reason, is that the theory will usually define behaviours through

the use of algebraic differential equations, and we conjecture that the sets of evolutions that can be constructed in this way are all finite set refutable. For algebraic equations, and for differential equations with unique solutions (Lipschitz equations, [HS74]), this is trivial. For differential equations in general, we only conjecture that finite set refutability holds on the basis of some topological closure and compactness properties that the solution sets of those equations have [Fil88].

However, we must be carefull in accepting these arguments, since they both make assumptions on the syntax. They might not hold anymore, if certain extensions of the syntax are introduced. For the moment, however, hybrid transition systems seem expressive enough for our needs, and seem to be the most suitable model from other points of view that were already mentioned. We expect that it will be possible, should the need to change semantics ever arise, to change the semantical model of a theory without disturbing the axioms and calculation rules on the syntax. Therefore, hybrid transition systems are the best model for the time being.

7 Hybrid Evolutions

Recall from section 3, that the notion of evolution is basic to theory on systems in general and in particular to both control and computer science. In that section, we have seen two ways in which evolutions are defined. Runs for transition systems, and functions of time for behavioural systems. Since a hybrid transition system is a transition system, the notion of run is most natural.

Definition 9 (Hybrid Run) *Given a hybrid transition system* $\langle X, \Sigma, T, \phi \rangle$, *a* hybrid run *of this system is a triple* (x, τ, σ) *of sequences* $x \in \mathbb{N} \mapsto X$, $\tau \in \mathbb{N} \mapsto T$, *and* $\sigma \in \mathbb{N} \mapsto ((T \mapsto \Sigma_C) \cup \Sigma_D)$ *such that*

- $Dom(x)$, $Dom(\tau)$, *and* $Dom(\sigma)$ *are intervals in* $\mathbb{N}$;
- $Dom(\tau) = Dom(x)$ *and* $0 \in Dom(x)$;
- $Dom(\sigma) \subseteq Dom(x) \ \wedge \ \forall_{n \in Dom(x)} \ n \in Dom(\sigma) \ \Rightarrow \ n+1 \in Dom(x)$;
- $\forall_{n \in Dom(\sigma)} \ \langle x(n), \tau(n) \rangle \stackrel{\sigma(n)}{\rightarrow} \langle x(n+1), \tau(n+1) \rangle$.

Notice that this definition has not fundamentally changed from the normal definition of run.

The notion of *run* as evolution of a system has one important drawback that manifests itself when runs become infinite in length. If there is a notion of

topology on the time axis and the state space of a system (think of T as $\mathbb{R}$ and of X as an arbitrary vector space), the intuition on what evolutions of the system are may change. In such a setting, there can exist runs for which the time-series does not pass a certain point $t \in T$. In such a case, using the definitions we have so far, the evolution of the system simply stops at, or before, that time. This is rather strange, since it will never happen in a physical system that time simply stops.

Usually, this phenomenon is caused by some abstraction made by the modeler. This does not mean that the model is wrong, or that the modeler has made a mistake! Abstraction is a tool that is crucial to the understanding of systems. In this particular case one may argue whether or not the modeler should be forced to avoid this kind of unwanted behaviour by specifying models that do not display the phenomenon, or one may decide that the modeling formalism should take care of it. We choose the latter because we are convinced that for example the assumption that certain actions do not take any time to execute is a natural abstraction. This assumption leads to the described phenomenon and therefore we should support this phenomenon in our modeling formalism.

A typical occurrence of the accumulation of a time-series is usually referred to as Zeno-behaviour after the Eleatic philosopher (488 BC) who first described such phenomena in his famous example of Achilles and the Turtle[4]. In philosophy the accumulation of events in general is also referred to as *supertask* [Zal01, Nor99, Sal70]. The state trajectory in such a case is crucial to the interpretation of the evolution of our system. Several attempts to deal with Zeno-behaviour in a hybrid systems context can be found in [JELS99, BP00, CRE01, CR02]. Usually it is assumed that the evolution of the system can continue from the accumulation point of states onward, forcing time to continue. This gives rise to transfinite sequences of states and to the notion of transfinite run.

A transfinite run is a run over ordinal numbers [Dug66, Eis74, Kun88] rather than natural numbers. The idea is that for limit ordinals the state-value of the run is one of the accumulation points of the preceding part of the run (if a sequence $x \in \Omega \to X$ accumulates in $y \in X$ we denote this by $x \multimap y$, see Definition 20 in appendix A). Also the timing should continue from the accumulation point of the previous timings.

Definition 10 (Transfinite Hybrid Run) *Let Ω denote the ordinal numbers and let $\langle X, \Sigma, T, \phi \rangle$ be a hybrid transition system with topologies on X and T, then a pair (x, τ, σ) of transfinite sequences $x \in \Omega \mapsto X$, $\tau \in \Omega \mapsto T$ and $\sigma \in \Omega \mapsto (\Sigma \cup (T \mapsto \Sigma))$ is a transfinite hybrid run if*

[4] In a running contest between Achilles the half-god and a turtle, the turtle gets a little head start. Zeno reasoned that every time Achilles comes at a point where the turtle was, the turtle has walked just a little further. Hence, Achilles cannot take over, and the turtle wins the race.

- $Dom(x)$, $Dom(\tau)$, *and* $Dom(\sigma)$ *are intervals in* Ω;
- $Dom(\tau) = Dom(x)$ *and* $0 \in Dom(x)$;
- $Dom(\sigma) \subseteq Dom(x) \ \wedge \ \forall_{n \in Dom(x)} \ n \in Dom(\sigma) \ \Rightarrow \ n+1 \in Dom(x)$;
- $\forall_{n \in Dom(\sigma)} \ \langle x(n), \tau(n) \rangle \stackrel{\sigma(n)}{\rightarrow} \langle x(n+1), \tau(n+1) \rangle$;
- $\forall_{n \in Dom(x), n \ limit \ ordinal} \ x|_{[0..n)} \multimap x(n) \ \wedge \ \tau|_{[0..n)} \multimap \tau(n)$.

Again, the length of a run is the cardinality of $Dom(\sigma)$.

Most results that have been obtained for hybrid systems in literature assume non-Zeno evolutions. This means that only normal runs are considered in the proofs. The problem of supertasks is considered a modeling mistake and hence left for the modeler to solve. In the remainder of this report, we explicitly make a distinction between results for normal hybrid runs and transfinite hybrid runs for this reason.

8 Equivalence

In sections 2 and 3 already the notion of *equivalence* was mentioned. Although not formally defined, the notion of bisimulation equivalence was given as an example.

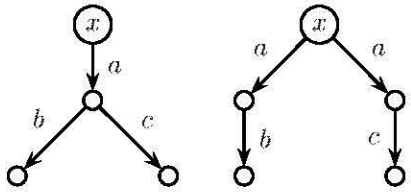

Figure 11: Trace Equivalent but not Bisimilar States

On a semantical level, the notion of equivalence on a system reflects which things an external observer can see *during* the evolution of a system. The first intuition we have about this equivalence, for systems in general, is that only the interaction space is observable. This intuition is firm, and all notions of equivalence from literature agree on it. The second intuition is motivated by the introduction of non-determinism in a system. If a system displays non-determinism, then sometimes a choice between different evolutions is made without visible interaction. See for example figure 11. There, the observer cannot conclude from the

set of runs of the system that the two states marked x are different. Both systems contain the interaction runs a followed by b and a followed by c. However, an observer can conclude difference from the fact that, after a has occurred, in one system there is still a choice between b and c, while in the other system this choice has vanished. This difference between observing complete runs and observing choices as well is only of interest for non-deterministic systems and therefore it is not surprising that the equivalence notions concerned with it stem from computer science rather than from control. If the outside observer cannot see which choices a system makes, we study so-called *trace equivalence* between systems. If the outside observer can observe the fact that a choice has been made, we study *bisimulation equivalence*. Equivalence, furthermore, is a notion on the states of systems and, in timed systems, this notion may be time dependent. Therefore, the equivalence notions we handle here compare pairs of state and time.

On a syntactical level, the chosen notion of equivalence influences greatly which calculation rules can be applied. Some axioms hold for one notion of equivalence, while they do not hold for a different notion. For example, with respect to trace equivalence we have that alternative composition ($+$) distributes over sequential composition ($\cdot$), leading to the axiom $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$, for those who are familiar with the process algebraic syntax. This is not true with respect to bisimulation equivalence! The choice of equivalence influences the analytical power of the theory greatly, because a notion of equivalence like bisimulation, that allows less axioms, has a greater distinctive power between systems. Therefore, one can pose more fine-grained questions about systems in the theory (for example about the choices that have been made). However, the fact that there are less axioms also means that answering questions by means of axiomatic reasoning becomes more difficult. So the questions that can be easily posed and answered for a less fine-grained equivalence, may turn out to be harder to answer for the fine-grained equivalence. This is why the choice of a proper notion of equivalence is of importance to the analyst.

Definition 11 (Trace Equivalence) *Given the hybrid transition systems* $M = \langle X_1, \Sigma_1, T_1, \phi_1 \rangle$ *and* $N = \langle X_2, \Sigma_2, T_2, \phi_2 \rangle$*, using the same interaction space* $\Sigma_1 = \Sigma_2$*, the state-time pairs* $(x_0, t_0) \in X_1 \times T_1$ *and* $(y_0, s_0) \in X_2 \times T_2$ *are* trace equivalent *denoted* $(x_0, t_0) \simeq (y_0, s_0)$*, if for every hybrid run* (x, t, σ) *on* M *with* $x(0) = x_0$ *and* $t(0) = t_0$*, there exists a hybrid run* (y, s, σ) *on* N *with* $y(0) = y_0$ *and* $s(0) = s_0$ *and vice versa.*

The notion of bisimulation equivalence does not compare complete runs, but compares the states after every transition again in order to be able to differentiate between the choices between transitions. It relies on the notion of bisimulation relation.

Definition 12 (Bisimulation Equivalence) *Given hybrid transition systems* $M = \langle X_1, \Sigma_1, T_1, \phi_1 \rangle$ *and* $N = \langle X_2, \Sigma_2, T_2, \phi_2 \rangle$ *with* $\Sigma_1 = \Sigma_2$. *A relation* $\mathcal{R} \subseteq (X_1 \times T_1) \times (X_2 \times T_2)$ *is a* bisimulation relation *iff*

- *for every transition* $\langle x, t \rangle \xrightarrow{\sigma} \langle x', t' \rangle$ *of* M *and every state* $y \in X_2$ *and time* $s \in T_2$ *such that* $(x, t)\ \mathcal{R}\ (y, s)$ *there exists a transition* $\langle y, s \rangle \xrightarrow{\sigma} \langle y', s' \rangle$ *of* N *such that* $(x', t')\ \mathcal{R}\ (y', s')$*;*
- *and for every transition* $\langle y, s \rangle \xrightarrow{\sigma} \langle y', s' \rangle$ *of* N *and every state* $x \in X_1$ *and time* $t \in T_1$ *such that* $(x, t)\ \mathcal{R}\ (y, s)$ *there exists a transition* $\langle x, t \rangle \xrightarrow{\sigma} \langle x', t' \rangle$ *of* M *such that* $(x', t')\ \mathcal{R}\ (y', s')$.

Two state-time pairs $(x, t) \in X_1 \times T_1$ *and* $(y, s) \in X_2 \times T_2$ *are bisimilar, denoted* $(x, t) \leftrightarroweq (y, s)$*, iff there exists a bisimulation relation* $\mathcal{R}$ *such that* $(x, t)\ \mathcal{R}\ (y, s)$. *(Two systems are bisimilar if every state in one system has a bisimilar state in the other system.)*

It is a known result from computer science [BPS01] that bisimulation on states is a stronger equivalence than trace equivalence on states.

Lemma 1 $\leftrightarroweq \subseteq \simeq$.

Proof See e.g. [vG01]. ⊠

Clearly, with topologies on the state spaces, extending the notion of trace equivalence is straightforward. Transfinite trace equivalence is concerned with transfinite runs instead of normal runs. It is denoted by $\simeq_\infty$. The bisimulation case, however, is not that simple. The notion of bisimulation focusses on single transitions and is therefore not able to "see" beyond a countable number of transitions. In [CR02], a notion of topological bisimulation was defined to include the accumulation points of runs. The following definition is a reformulation of this definition.

Definition 13 (Topological Bisimulation) *Let* $M = \langle X_1, \Sigma_1, T_1, \phi_1 \rangle$ *and* $N = \langle X_2, \Sigma_2, T_2, \phi_2 \rangle$ *be two hybrid transition systems such that* $\Sigma_1 = \Sigma_2$ *and such that* M *and* N *have topologies on the state spaces* X_1 *and* X_2 *and on the time axes* T_1 *and* T_2*, the relation* $\mathcal{R} \subseteq (X_1 \times T_1) \times (X_2 \times T_2)$ *is a* topological bisimulation relation *iff it is a (normal) bisimulation relation that also relates accumulation points of transfinite hybrid runs, i.e.:*

1. *for every transfinite hybrid run* (x, τ, σ) *in* M *and state* $y_0 \in X_2$ *such that* $(x(0), \tau(0)) \mathcal{R} (y_0, \tau(0))$*, there is a transfinite hybrid run* (y, τ, σ) *in* N *with* $y_0 = y(0)$ *and* $(x(n), \tau(n)) \mathcal{R} (y_n, \tau(n))$ *for every* $n \in Dom(x)$.

2. *for every transfinite hybrid run* (y, τ, σ) *in* N *and state* $x_0 \in X_1$ *such that* $(x_0, \tau(0))\mathcal{R}(y(0), \tau(0))$, *there is a transfinite hybrid run* (x, τ, σ) *in* M *with* $x_0 = x(0)$ *and* $(x(n), \tau(n))\mathcal{R}(y_n, \tau(n))$ *for every* $n \in Dom(y)$.

A state-time pair $(x, t) \in X_1 \times T_1$ *is topologically bisimilar to a state* $(y, s) \in X_2 \times T_2$, *denoted* $(x, t) \leftrightarrows_\infty (y, s)$, *if and only if there exists a topological bisimulation relation* $\mathcal{R} \subseteq (X_1 \times T_1) \times (X_2 \times T_2)$ *such that* $(x, t)\ \mathcal{R}\ (y, s)$.

As with normal bisimulation and trace equivalence, topological bisimulation is stronger than transfinite trace equivalence.

Lemma 2 $\leftrightarrows_\infty \subseteq \simeq_\infty$.

Proof The proof of this is trivial, since the equivalence notion, for every transfinite hybrid run in the one system, guarantees the existence of a similarly labelled transfinite hybrid run in the other. ⊠

Now we have covered evolution and equality, the most important notions on systems from a semantical point of view. In the next section, we look at control theory, and sketch the way in which standard control notions can be defined on hybrid transition systems.

9 Control Notions

The notion of evolution is the most important notion in order to support a general intuition on any type of system. Equivalence notions are invaluable if we want to consider hybrid transition systems as an algebraic structure. However, other notions are important from a practical point of view. In this section, we sketch the definitions of a few notions from control theory as an example of how this theory can be incorporated into the hybrid structure. It is not our intention yet to prove properties about those notions. This is left as future research.

Most of the theoretical notions we are interested in depend in one way or another on the notions of evolution and equivalence. For example, the notion of time-invariance states that, given a certain notion of equivalence, the equivalence of two states is independent of time. In the remainder of this report, we write $\stackrel{\circ}{=}$ to denote an arbitrary notion of equivalence that has been chosen by the modeler. In place of $\stackrel{\circ}{=}$ one may read any of the equivalences $=$, $\simeq$, $\simeq_\infty$, $\leftrightarrows$, or $\leftrightarrows_\infty$ as desired.

Definition 14 ($\stackrel{\circ}{=}$-Time-invariance) *We define that a hybrid transition system $\langle X, \Sigma, T, \phi \rangle$ is* time-invariant *with respect to a certain notion of equivalence $\stackrel{\circ}{=}$ iff*

$$\forall_{x \in X} \forall_{t,t' \in T} (x, t) \stackrel{\circ}{=} (x, t').$$

Note that equivalence of states is here considered between the states of one single system (M and N are the same). For x and y states from hybrid transition systems M and N, we write $x \stackrel{\circ}{=} y$ if $(x, t) \stackrel{\circ}{=} (y, s)$ for all t and s.

The assumption of time-invariance usually makes modeling and analysis of systems easier. Also, many definitions from control theory assume a time-invariant system. We therefore use this notion extensively in this section. In classical theories, time-invariance is usually guaranteed if one only uses syntactical terms that do not refer to time immediately. Of course, in future investigations, we strive for a similar result for hybrid syntactical terms.

Arguably, two of the most important notions from control theory are *observability* and *controllability* of the state of a system. Observability roughly means that one can know in which state the system is from observing the interactions that have taken place. Controllability means that it is possible to steer the system from any state into any other state, by applying the right interactions.

Definition 15 ($\stackrel{\circ}{=}$-Observability) *Let $\langle X, \Sigma, T, \phi \rangle$ be a hybrid transition system that is time-invariant with respect to equivalence $\stackrel{\circ}{=}$, then the state $y \in X$ is observable from the interaction sequence $\sigma \in \Omega \mapsto ((T \mapsto \Sigma_C) \cup \Sigma_D)$ of length n if for every run (x, τ, σ) we find $x(n) \stackrel{\circ}{=} y$.*

This definition simply says that if we observe that the interaction σ has occurred it is safe to conclude that the state after this observation is equivalent to y. Please notice again that this notion depends on the chosen notion of equivalence. It maybe so that with respect to a weaker notion of equivalence a system is observable while it is not with respect to a stronger notion. This happens because the states that can be reached by the observed interaction are equal with respect to the weaker notion while they were not equal with respect to the stronger notion. In standard control theory this distinction is not made because the determinism assumed there makes that bisimulation and trace equivalence coincide, but from a computer science point of view, where non-determinism does play a role, it is important. Also important, is the so called congruence property of a notion with respect to a chosen equivalence. For example, suppose we have two equivalent systems $M \stackrel{\circ}{=} N$, then we expect if a certain state x is observable from σ in M that there is an equivalent state $y \stackrel{\circ}{=} x$ observable from σ in N. It is a subject of future investigation whether observability is a congruence for the notions of equivalence that have been given in the previous sections.

Controllability is a notion that expresses if a system can be steered from one state into another.

Definition 16 ($\stackrel{\circ}{=}$-Controllability) *Let $\langle X, \Sigma, T, \phi \rangle$ be a hybrid transition system that is time invariant with respect to $\stackrel{\circ}{=}$, then the state $y \in X$ can be* controlled *into state $y' \in X$ by the interaction $\sigma \in \Omega \mapsto ((T \mapsto \Sigma_C) \cup \Sigma_D)$ with respect to the equivalence $\stackrel{\circ}{=}$, if there exists a (transfinite) hybrid run (x, τ, σ) such that $x(0) = y$ and $x(n) \stackrel{\circ}{=} y'$ for some n.*

For control scientists who usually work with deterministic systems, this notion works perfectly. When non-determinism comes into play, however, it may seem rather naive. The fact that there is a run from y to y' using a certain interaction is no guarantee that this particular path of states is indeed chosen when the particular interaction is applied. However, demanding a guarantee that after the interaction we indeed end up in y' is similar to demanding a degree of determinism. On the other hand, it may be reasonable to assume that after a failed attempt the controller will be able to try again. The assumption that y' is eventually reached if there is always a run leading to it, is called *fairness* in computer science.

It is clear that this notion of controllability is suitable for control theory, but does not cover the intuition anymore when applied to hybrid systems. Future research will have to come up with new definitions of controllability that fit the non-deterministic framework.

Also for controllability, the aforementioned congruence property is important. In particular, we do not have the desired congruence if we consider for example the combination of transfinite hybrid runs and normal bisimulation. Trivially, the (normal bisimulation) equivalence of systems M and N does not guarantee that if, for example, every state in M is (transfinitely) controllable into every other, that also every state in N is controllable into every other.

The last control notion that we want to address here is called stability of a system. Stability is a notion concerned with the desire to guarantee that the behaviour of a system remains within reasonable bounds. In literature, many different notions of stability have been given. One example from this large pool of possibilities is the notion of a bounded state evolution. A system is stable if all the state evolutions stay within a certain bound. To be able to define this, the topology on the state space and signal space is assumed to be induced by a metric (see definition 21 in appendix A). In literature, many other notions of stability have been given, most of them have the same topological nature.

Definition 17 (Stability) *A hybrid transition system with a metric on the state space X and signal space Σ is stable if for every run (x, τ, σ) in which σ is bounded, we find that x is bounded (see definition 22 in appendix A).*

10 Concluding Remarks and Future Research

We have provided a semantics for studying hybrid systems by introducing the modeling framework of hybrid transition systems as a mix of Sontag machines and timed labeled transition systems. This framework was compared with the original frameworks from control [Son98, PW98, TSH01] and computer science [BM01], as well as with hybrid automata [LSV99, LSV01] and the notion of rich time in a behavioural setting [vdSS00b]. We have shown how the notions from computer science and control science can be incorporated into the new framework by giving a few examples of definitions of such notions in terms of hybrid transition systems. Furthermore, we discussed the influence of topology on computer science notions and in particular the extension of classical notions of equivalence to overcome Zeno-behaviour, a typical hybrid problem induced by topology.

Our arguments for choosing hybrid transition systems above hybrid automata or behavioural systems are based on our feeling that hybrid automata have a mathematically awkward structure while hybrid behavioural systems are to much focussed on the complete evolutions of systems. We prefer an operational, state-based, view on systems because the possibility of non-determinism (in particular in computer science models) urges us to look beyond the evolution based language equivalence of systems. The fact has been recognized, that hybrid transition systems are not expressive enough, compared with behavioural systems, to distinguish between continuous moments of choice and moments of choice that are at arbitrarily small times apart. This lack in expressivity has been rejected as a potential cause of trouble because the information that is lost, is not used in any of the compositions that we have in mind on hybrid systems. Furthermore, most behavioural systems that are currently of interest, for example those defined by algebraic and differential equations, have the property of *finite set refutability*, which ensures that there is no problem in the translation to hybrid transition systems.

Our hopes are that the proposed semantics will provide a sufficient basis for the development of an algebra, suitable for the analysis of hybrid systems, with respect to the notions mentioned in this report. The challenge will be to find axioms and theorems on the notions we know so well from classical theory, in a hybrid context. The future development of theory should, apart from syntax, include theorems about the control notions that were mentioned. In particular, the congruence of those notions with respect to the different kinds of equivalence is important. From a semantical viewpoint, it would be nice if those control notions could be derived from the transition systems itself, rather than having definitions via the runs. This would add to the algebraic simplicity of the notions, and, through that, probably also to stronger syntactical theorems.

Acknowledgements: We would like to thank Paul van den Bosch, Jan Friso Groote, Aleksandar Juloski, Ka Lok Man, Ramon Schiffelers, Tim Willemse, Marc Voorhoeve, Erik Gaal and Arno de Vet, for their comments during a few intensive sessions on mathematical modeling in general and semantics in particular. Explaining things to a critical yet willing audience is often a crucial prerequisite for progress. Furthermore, we would like to express our gratitude to Progress/STW, Philips-CFT and Assembleon, for their financial and material support to our project.

References

[AD94] R. Alur and D.L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[Are96] N.W.A. Arends. *A Systems Engineering Specification Formalism*. PhD thesis, TU/e, June 1996.

[BB91] J.C.M. Baeten and J.A. Bergstra. Real time process algebra. *Formal Aspects of Computing*, 3(2):142–188, 1991.

[BBM98] M.S. Branicky, V.S. Borkar, and S.K. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE Transactions on Automatic Control*, 43(1):pp. 31–45, January 1998.

[BK02] V. Bos and J.J.T. Kleijn. *Formal specification and analysis of industrial systems*. PhD thesis, TU/e, 2002.

[BM99] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics and constraints. *Automatica*, 35(3):407–427, 1999.

[BM01] J.C.M. Baeten and C.A. Middelburg. Process algebra with timing. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, chapter 10, pages 627–684. Elsevier Science B.V., Amsterdam, 2001.

[BM02] J.C.M. Baeten and C.A. Middelburg. *Process Algebra with Timing*. Monographs in Theoretical Computer Science. Springer-Verlag, 2002.

[BP00] B. Bérard and C. Picaronny. Accepting zeno words: a way toward timed refinements. *Acta Informatica*, 37(1):45–81, 2000.

[BPS01] J.A. Bergstra, A. Ponse, and S.A. Smolka, editors. *Handbook of Process Algebra*. Elsevier Science B.V., Amsterdam, 2001.

[BW90] J.C.M. Baeten and W.P. Weijland. *Process Algebra*, volume 18 of *Cambridge Trancts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 1990.

[CR02] P.J.L. Cuijpers and M.A. Reniers. Topological (bi-)simulation. Technical Report CS-Report 02-04, TU/e, Eindhoven, Netherlands, 2002.

[CRE01] P.J.L. Cuijpers, M.A. Reniers, and A.G. Engels. Beyond zeno-behaviour. Technical Report CS-Report 01-04, TU/e, Eindhoven, Netherlands, 2001.

[DMY02] A. David, M.O. Möller, and W. Yi. Formal verification of UML statecharts with real-time extensions. In R.D. Kutsche and H. Weber, editors, *Fundamental Approaches to Software Engineering*, volume 2306 of *LNCS*, pages 218–232. Springer-Verlag, 2002.

[Dug66] J. Dugundji. *Topology*. Allyn and Bacon, Inc., Boston, 1966.

[EBF+98] A. Evans, J. Bruel, R. France, K. Lano, and B. Rumpe. Making UML precise. In L. Andrade, A. Moreira, A. Deshpande, and S. Kent, editors, *Proceedings of the OOPSLA'98 Workshop on Formalizing UML. Why? How?*, 1998.

[Eis74] M. Eisenberg. *Topology*. Holt, Rinehart and Winston, Inc., New York, 1974.

[Fil88] A.F. Filippov. *Differential Equations with Discontinuous Right-hand Sides*. Mathematics and its applications (Soviet series). Kluwer Academic Press, 1988.

[Fok98] W. Fokkink. *Introduction to Process Algebra*. Texts in Theoretical Computer Science. Springer-Verlag, Berlin, 1998.

[GP95] J.F. Groote and A. Ponse. The syntax and semantics of μCRL. In A. Ponse, C. Verhoef, and S.F.M. van Vlijmen, editors, *ACP: Algebra of Communicating Processes*, Workshops in Computing, pages 26–62, Utrecht, 1995. Springer-Verlag.

[GPP98] M. Gogolla and F. Parisi-Presicce. State diagrams in UML: A formal semantics using graph transformations. In M. Broy, D. Coleman, T.S.E. Maibaum, and B. Rumpe, editors, *Proceedings PSMT'98 Workshop on Precise Semantics for Modeling Techniques*. Technische Universität München, TUM-I9803, 1998.

[GR01] J.F. Groote and M.A. Reniers. Algebraic process verification. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, chapter 17, pages 1151–1208. Elsevier Science B.V., Amsterdam, 2001.

[Hen96] T.A. Henzinger. The theory of hybrid automata. In *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS 1996)*, pages 278–292. IEEE Computer Society Press, 1996.

[HS74] M.W. Hirsch and S. Smale. *Differential Equations, Dynamical Systems, and Linear Algebra.* Pure and Applied Mathematics. Academic Press, 1974.

[JELS99] K.H. Johansson, M. Egerstedt, J. Lygeros, and S. Sastry. On the regularization of zeno hybrid automata. *System and Control Letters*, 38:141–150, 1999. Special issue on Hybrid Systems.

[Kun88] K. Kunen. *Set Theory: An Introduction to Independence Proofs*, volume 102 of *Studies In Logic and the Foundations of Mathematics.* Elsevier Science B.V., third edition, 1988.

[LSV99] N. Lynch, R. Segala, and F. Vaandrager. Hybrid I/O automata. Technical Report CSI-R9907, University of Nijmegen, 1999.

[LSV01] N. Lynch, R. Segala, and F. Vaandrager. Hybrid I/O automata revisited. In M.D. Di Benedetto and A.L. Sangiovanni-Vincentelli, editors, *Proceedings Fourth International Workshop on Hybrid Systems: Computation and Control (HSCC'01)*, volume 2034 of *Lecture Notes in Computer Science*, pages 403–417. Springer-Verlag, 2001.

[Mil80] R. Milner. *A calculus of communicating systems*, volume 92 of *Lecture Notes in Computer Science.* Springer-Verlag, 1980.

[Nor99] J.D. Norton. A quantum mechanical supertask. *Foundations of Physics*, 29:1265–1302, 1999.

[Phi01] P.P.H.H. Philips. *Modelling, Control and Fault Detection of Discretely-Observed Systems.* PhD thesis, TU/e, 2001.

[PW98] J.W. Polderman and J.C. Willems. *Introduction to Mathematical Systems Theory: A Behavioural Approach*, volume 26 of *Texts in Applied Mathematics.* Springer-Verlag, 1998.

[RGvdZvW02] M.A. Reniers, J.F. Groote, M.B. van der Zwaag, and J. van Wamel. Completeness of timed μCRL. *Fundamenta Informaticae*, 50(3-4):361–402, 2002.

[Sal70] W. Salmon, editor. *Zeno's Paradoxes.* Bobbs-Merril, Indianapolis, 1970.

[Son98] E.D. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems*, volume 6 of *Texts in Applied Mathematics.* Springer-Verlag, 1998.

[TSH01] H.L. Trentelman, A.A. Stoorvogel, and M. Hautus. *Control Theory for Linear Systems.* Springer-Verlag, 2001.

[vBGR97] D.A. van Beek, S.H.F. Gordijn, and D.E. Rooda. Integrating continuous-time and discrete-event concepts in modelling and simulation of manufacturing machines. *Simulation Practice and Theory*, 5:653–669, 1997.

[vBR00] D.A. van Beek and J.E. Rooda. Languages and applications in hybrid modelling and simulation: Positioning of χ. *Control Engineering Practice*, 8(1):81–91, 2000.

[vdB98] P.P.J. van den Bosch. Hybrid systems: modelling embedded controllers (invited paper). In *Proceedings Philips conference on Applications of control technology*, pages pp. 1–8, Groenendaal, November 1998.

[vdSS00a] A.J. van der Schaft and J.M. Schumacher. Compositionality issues in discrete, continuous, and hybrid systems. Technical Report Memorandum 1549, University of Twente, Enschede, Netherlands, 2000.

[vdSS00b] A.J. van der Schaft and J.M. Schumacher. *An Introduction to Hybrid Dynamical Systems*, volume 251 of *Lecture Notes in Control and Information Sciences.* Springer-Verlag, London, 2000.

[vG01] R.J. van Glabbeek. The linear time – branching time spectrum I: The semantics of concrete, sequential processes. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, chapter 1, pages 3–99. Elsevier Science B.V., Amsterdam, 2001.

[Wei91] S. Weiland. *Theory of Approximation and Disturbance Attenuation for Linear Systems.* PhD thesis, Rijksuniversiteit Groningen, 1991.

[Zal01] E.N. Zalta, editor. *Stanford Encyclopedia of Philosophy.* The Metaphysics Research Lab, CSLI, Stanford University, 2001. http://plato.stanford.edu.

A Topological Notions and Notations

In this section we give a very brief explanation of some of the topological notions and notations used in this report. For a more thorough explanation we refer to [Dug66, Eis74].

We start out with the notion of total ordering, and the derived notion of interval, which turn out to be useful notions on the time axis T of a system.

Definition 18 (Total ordering, Interval) *Let T be a set and $\leq \subseteq T \times T$ a (binary) relation on T. This relation is called a* total ordering, *and T is called* totally ordered *if we have for all $a, b, c \in T$:*

- *reflexivity:* $a \leq a$;
- *antisymmetry:* $a \leq b \ \wedge \ b \leq a \ \Rightarrow \ a = b$;
- *transitivity:* $a \leq b \ \wedge \ b \leq c \ \Rightarrow \ a \leq c$;
- *totality:* $a \leq b \ \vee \ b \leq a$;

A subset $I \subseteq T$ of a totally ordered set is called an interval *if for all $x, y, z \in T$*

- *interval:* $x, z \in I \ \wedge \ x \leq y \leq z \ \Rightarrow \ y \in I$.

A different kind of structuring on sets is through a topology. Roughly, a topology defines which points in a set are close to each other. A set equipped with a topology is called a (topological) space.

Definition 19 (Topology) *Let X be a set, then $\mathcal{T} \subseteq \mathcal{P}(X)$ is a* topology *on X if*

- $\emptyset \in \mathcal{T}$ *and* $X \in \mathcal{T}$;
- *all unions of elements of $\mathcal{T}$ are itself elements of $\mathcal{T}$;*
- *all finite intersections of $\mathcal{T}$ are itself elements of $\mathcal{T}$.*

The elements $o \in \mathcal{T}$ are called open sets. *For a point $x \in X$, an open set containing x is called a* neighbourhood *of x.*

A trajectory through a topological space is said to accumulate at x if every neighbourhood of x is visited over and over again. I.e. the trajectory gets arbitrarily close to x, but may get away from x from time to time as well. Formally we define this as follows.

Definition 20 (Accumulation) *Let A be a set totally ordered*[5] *by $\leq$ and X be a set with topology $\mathcal{T}$, then a function $f \in A \to X$ is* accumulating *at $x \in X$, denoted $f \multimap x$, if*

$$\forall_{o \in \mathcal{T}, x \in o} \forall_{a \in Dom(f)} \exists_{b \in Dom(f), b \geq a} \ f(b) \in o \ .$$

[5] In the actual topological definitions, $\leq$ is only required to be a directed pre-order, not necessarily a total order. However going into details about this difference is not necessary for the understanding of this report.

Accumulation gives us an intuition about the limit points of sequences. A set in which every limit point is also a part of the set, is called topologically closed.

A special way to induce a topology is through a metric. A metric is a function that defines the distance between two points. Naturally distance is one way of defining how close points are to each other.

Definition 21 (Metric) *Let X be a set, then a metric on that set is a function $||.,.|| \in (X \times X) \to \mathbb{R}$ that is*

- *positive:* $\forall_{x,y \in X} ||x, y|| \geq 0$;
- *distinctive:* $\forall_{x,y \in X} ||x, y|| = 0 \Leftrightarrow x = y$;
- *symmetric:* $\forall_{x,y \in X} ||x, y|| = ||y, x||$;
- *triangular:* $\forall_{x,y,z \in X} ||x, z|| \leq ||x, y|| + ||y, z||$.

Every metric induces a topology $\mathcal{T} \subseteq \mathcal{P}(X)$ such that for all $x \in X$ and $d \in \mathbb{R}$:

$$\{x' \mid ||x, x'|| < d\} \in \mathcal{T}.$$

Metrics also induce a notion of boundedness, reflecting that a certain distance is never exceeded.

Definition 22 (Bounded function) *Let X be a set with a metric on it. A function $f \in \mathbb{N} \mapsto X$ is* bounded *if*

$$\exists_{d \in \mathbb{R}} \forall_{n,m \in \mathbb{N}} \; ||f(n), f(m)|| \leq d \; .$$