First-Order Logic Models for Real-Time, Discrete-Event Systems

Arch Naylor
University of Michigan

May, 1993

Abstract

A methodology based on first-order logic for modeling discrete-event systems is introduced. Time is the real line, and systems are allowed to have an infinite number states. Applications of the modeling methodology are presented.

1. Introduction

Our purpose is to introduce a modeling methodology for discrete-event systems [1, 2, 3, 4] and to demonstrate its use. The methodology is based on first-order logic [5]. We use logic because most discrete-event systems are naturally characterized by logical conditions and quantities, and we choose first-order logic because it is by far the most widely known and developed version of logic. And, first-order logic is, as we will show, expressive enough to model the phenomena of interest. In particular, we can have real time, simultaneous events, and infinite state sets, yet still obtain classes of tractable models.

However, even though first-order logic is the best known version of logic, few know more than introductory concepts, and setting up our modeling methodology requires more. Fortunately, *using* the methodology does not. Here we will cover the background technical issues but in a largely segregated manner. Thus, a reader willing to accept things on faith can skip over their discussion.

Another possibility would have been to do everything semantically in terms of sets, ignoring syntax and the connections to logic. However, we would still have to reason about these sets, and that would mean we were back to logic, but in an informal, somewhat disorganized, manner. We believe an orderly theoretical foundation is better. It allows us to use known facts of first-order logic. For example, Beth's theorem [6, page 87] is very useful.

The modeling formalism yields models that are systems of what we call pseudo-differential-difference equations[1]. They are analogous to systems of ordinary differential difference equations, and they are used in the same way. That is, one tries to find closed-form solutions where they exist; one deduces properties of solutions from the form of the system of equations; one uses approximations such as linearization, and one exercises the system of equations in simulations.

2. Dynamic and Static Quantities

We view a discrete-event system as being made up of dynamic and static quantities. The dynamic quantities are predicate-valued and function-valued functions of time.

Example 1 *Time is the real line $\Re$. Vehicles are in a set V of ten vehicles. There are twenty locations in a set L. The predicate $Loc\_(\tau,v,\ell)$, that is, some subset of the Cartesian product $\Re \times V \times L$, keeps track of vehicle location. We view $Loc\_(\tau,v,\ell)$ as a mapping of $\Re$ into $2^{V\times L}$, the power set of $V \times L$. That is, we view it as a predicate-valued function of time. For example, if $< 1, v_1, \ell_7 >$ and $< 1, v_3, \ell_2 >$ are the only ordered triples in $Loc\_$ with $\tau = 1$, then the value of $Loc\_$ at time 1 is the predicate $\{< v_1, \ell_7 >, < v_3, \ell_2 >\}$. Vehicle 1 is at location 7, vehicle 3 is at location 2, and no other vehicle is at a location; that is, they are between locations.*

Example 2 *In Example 1 the predicate $Loc\_(\tau, v, \ell)$ might contain both $< 1, v_1, \ell_8 >$ and $< 1, v_1, \ell_{12} >$, that is, it might specify that vehicle 1 is at two different locations at the same time. In some systems this might not be physically possible, and the model would presumably have to reflect this constraint.*

One way to reflect it is to require that the sentence

$$(\forall\tau)\psi(\tau) = (\forall\tau)(\forall v)(\forall\ell)(\forall\ell')\,[Loc\_(\tau,v,\ell) \wedge Loc\_(\tau,v,\ell') \rightarrow (\ell = \ell')] \tag{1}$$

be satisfied, that is, if v is at ℓ and ℓ' at time τ, then ℓ and ℓ' have to be the same. Using this approach would require that we would have to show that ψ was satisfied initially and that no subsequent event causes $\psi(\tau)$ not to be satisfied. This is a reasonable approach, and there are many situations in which this technique is used. Indeed, it is a simple example of supervisory control which we discuss later.

Example 3 *Another approach to the problem addressed in Example 2 is to replace the predicate $Loc\_(\tau, v, \ell)$ by a function $Loc\_(\tau, v)$, that is, by a mapping of $\Re \times V$ into L. $Loc\_(\tau,v)$ can be viewed as a function-valued function of time; in particular, at any fixed*

[1] A related approach is presented in [7].

time τ, $Loc\_(\tau,v)$ yields a mapping of V into L. This mapping assigns a unique location to each vehicle at time τ. Since $Loc\_$ is a function, we do not need sentence 1. On the other hand, since $Loc\_$ is defined on all of $\Re \times V$, a vehicle is always at some location, that is, a vehicle cannot be between locations.

We will organize dynamic quantities into dynamic predicate families and dynamic function families. A dynamic predicate family is an ordered pair of predicates $\{P\_(\tau, x_1, ...x_n), \delta P\ (\tau, x_1, ..., x_n)\}$. We call this a family because, as we will explain below, δP is supposed to be what we call the pseudo-derivative of $P\_$. We say "supposed to be" because it is not always the case for technical reasons which we will also explain below. Similarly, a dynamic function family is an ordered pair $\{F\_(\tau, x_1, ..., x_n), \delta F(\tau, x_1, ..., x_n, y)\}$, where $F\_$ is a function and δF is a predicate. δF is supposed to be the pseudo-derivative of the predicate defined by $(y = F\_\ (\tau, x_1, ..., x_n))$.

Quantities that have no occurrence of time are said to be static. Further, and a bit inconsistently, some quantities with occurrences of time are still classified as static. For example, addition of two times certainly has occurrences of times, but we say that this addition is static. Dynamic quantities are really things that are dynamic in the discrete-event system being modeled.

Each dynamic family is classified as either endogenous or exogenous. The endogenous families are inside the system and the exogenous ones are inputs.

3. $\mathcal{L}$, Language

Each model of a discrete-event system is expressed in terms of a first-order language $\mathcal{L}$ [5, page 67] that is appropriate for the system being modeled. Such a language has variable and constant symbols, predicate and function symbols, and logic symbols. There are the usual syntactic rules for combining these symbols into well-formed formulas [5, page 73] and sentences [5, page 75]. The language is "appropriate" in the sense that its constant, predicate, and function symbols are associated with entities and concepts of interest in the system being modeled. Each model of a discrete- event system is a collection of sentences over a language $\mathcal{L}$.

Remark 1 *Logic has a syntactic and a semantic side. An example of the syntactic side is a predicate symbol $Loc\_$, and an example of the semantic side is a predicate that provides a meaning or interpretation for the symbol $Loc\_$. Predicates $P_1\_(\tau,\ v,\ \ell)$ and $P_2\_(\tau,\ v,\ \ell)$ might be two different interpretations. Thus, there are* predicate symbols *and* predicates. *Often authors will use different notations for these two concepts, for example, $Loc\_$ for the symbol and $\widetilde{Loc}\_$ for a predicate. Here, however, we will risk a bit of confusion by using the same notation for both concepts.*

Remark 2 *An interpretation for a first-order language is a collection of interpretations of the constant, predicate, and function symbols over one or more universes of discourse. This is usually called a* structure *[5, page 79] for the language. There are many structures for any first-order language. A structure is said to* satisfy *[5, page 81] a first-order sentence α if α is a true statement about the structure. It satisfies a set of first-order sentences if it satisfies each sentence in the set.*

We will denote a model of a discrete-event system by $\sum$. The sentences in $\sum$ are subdivided into the following subsets:

- $\sum_T$, the sentences defining time
- $\sum_{PD}$, the sentences defining pseudo-differentation
- $\sum_{DEB}$, the sentences defining discrete-event behavior
- $\sum_{RS}$, the sentences defining the right sides of the pseudo-differential- difference equations
- $\sum_S$, the sentences defining other static quantities
- $\sum_\omega$, sentences added to support universal quantification over time

Initial conditions and inputs are specified by a set of sentences $\sum_{IC\&E_x}$. A set of sentences $\sum_H$ connects initial conditions to the original dynamic quantities. We will denote $\sum \cup \sum_{IC\&E_x} \cup \sum_H$ by $\sum_t$.

Remark 3 *The subsets $\sum_T, \sum_{PD}, \sum_{DEB}$ and $\sum_\omega$ are essentially common to all models. They will differ only because the underlying first-order languages differ. In a sense, this part of $\sum$ defines the mathematical environment of a model. Logical models are peculiar in that they contain everything that may be needed. In models using ordinary differential equations, one need not include a definition of differentiation as part of the model: it is implicit. In contrast $\sum_{PD} \subset \sum$. Of course, as long as $\sum$ is going to be worked with only "by hand", much of it can be made implicit, too. However, any automated use of $\sum$ will have to include everything in one way or another.*

Remark 4 *In the basic version of first-order logic, all the symbols in a structure are interpreted with respect to one universe. Thus, time and parts and vehicles and locations are mixed together into a common universe of discourse. This is sometimes awkward, so different categories are sometimes interpreted with their own separate universes of discourse. Thus, there would be a universe for times, another one for parts, another for vehicles, another for locations, and so on.*

To do this, one uses what is called a many-sorted first-order language [5, page 277]. For example $Loc\_(\tau,\ v,\ \ell)$ could be many sorted with τ a symbol of the sort time, v of the *sort vehicle, and ℓ of the sort location. A structure then requires a set T for times, a set V for vehicles, a set L for locations, and $Loc\_$ is a subset of $\Re \times V \times L$. If, instead, we did not use a many-sorted language, a structure would have a common universe U, and $Loc\_$ would be a subset of $U \times U \times U$. But we would need additional predicates to say what subsets of U were times, vehicles, and locations, respectively. For example, $T(x)$, $V(y)$ and $L(z)$ would define times, locations, and vehicles, respectively. Then we would need sentences saying that the defined sets were pairwise disjoint, for example, $(\forall x)(\neg(T(x) \wedge V(x)))$. And so on. In summary, then, many-sorted languages simplify things, so our models will use them. However, to simplify things even further, we usually let the many-sortedness be implicit.*

Example 4 *A transport system has one vehicle which moves from one location to the next when a move command is given. The route is a fixed loop. There are two sorts: time and locations. The endogenous dynamic families are $\{W\_(\tau, l), \delta W(\tau, l)\}$, and $\{S\_(\tau, l, \tau_a), \delta S(\tau, l, \tau_a)\}$. $W\_$ models "Where is the vehicle stopped now?" Its value at any time is either the empty set or a set containing a single location. The value of δW is empty at all time except those where a move starts or finishes. $S\_$ models an ongoing move with τ denoting the current time, l denoting the destination location, and τ_a denoting the scheduled arrival times. If there is no move currently in progress, the value of $S\_$ will be the empty set. The value of δS at any time is empty except at the start and finish of moves. The single exogenous dynamic family is $\{M\_(\tau), \delta M(\tau)\}$, which models move requests. This model has an infinite state set because τ_a can be any real number.*

4. Σ_T, Time

Time is the real line $\Re$.

Remark 5 *However, there is a technical issue. Our modeling methodology is developed* within *first-order logic, but we want time to be the real line. Unfortunately, we cannot have both. If we insist on defining time as exactly the real line, then we have to go beyond first-order logic, and if we insist on staying within first-order logic, then we cannot define exactly the real line. Let us see why.*

First-order logic allows predicates, functions, and quantification of only variables which refer to elements of the universe of discourse, for example, $(\forall x)$ and $(\exists x)$. Second-order logic allows predicate and function variables, and it allows quantification of these variables, for example, $(\forall X)$, $(\exists X)$, $(\forall F)$, and $(\exists F)$. That is, we can make statements such as "there exists a set ..." and "there exists a function ...". Higher-order logics allow us to talk about sets of sets, sets of sets of sets, and so on.

If one looks at a serious definition of the real line in any graduate analysis book, one discovers, once the definition has been translated into an appropriate logical formulation, that the definition uses some higher-order constructs, and, it turns out, that they are unavoidable.

Indeed, let $\Gamma_{\Re}$ be the set of all first-order sentences that are true statements about the real line. $\Gamma_{\Re}$ is obviously consistent because $\Re$ satisfies it. Since any first-order sentence α is either satisfied by the real line or not satisfied, $\Gamma_{\Re}$ is complete [5, page 145]. Complete means that for any first-order sentence α either $\alpha \in \Gamma_{\Re}$ or $\alpha \notin \Gamma_{\Re}$ and not both. In other words, as long as "first-order questions" are posed, $\Gamma_{\Re}$ contains all the answers, However, if we ask a second- or higher-order question, $\Gamma_{\Re}$ may not allow us to determine the answer.

Suppose that β is a higher-order sentence saying something true about the real line. For example, let β be

$$(\forall X)\,[\{(\exists y)(\forall z)\,(X(z) \rightarrow (z < y)) \wedge (\exists z)X(z)\} \rightarrow \\ \{(\exists y)(\forall y')\,((\forall z)(X(z) \rightarrow (z < y')) \leftrightarrow (y \leq y')\}] \tag{2}$$

where X is a unary predicate variable symbol. This second-order sentence says that any bounded nonempty set has a least upper bound. β is certainly a true statement about the real line, but it does not follow from $\Gamma_{\Re}$, that is, there are structures that satisfy $\Gamma_{\Re} \cup \{\beta\}$ (e.g., $\Re$) and other structures that satisfy $\Gamma_{\Re} \cup \{\neg\beta\}$. This means that neither β nor $\neg\beta$ can be deduced [2] from $\Gamma_{\Re}$. β is an important property of the real line, then, that cannot be expressed in first-order logic. Consequently, our first-order version of time will not be able to talk about least upper bounds, among other things.

Since there are structures satisfying $\Gamma_{\Re} \cup \{\neg\beta\}$ and $\Re$ does not satisfy $\Gamma_{\Re} \cup \{\neg\beta\}$, there are structures satisfying $\Gamma_{\Re}$ that are not isomorphic to $\Re$. Such a structure is usually referred to as a nonstandard model[3] *of the real line. The point is that there are structures which satisfy $\Gamma_{\Re}$ that do not "look like" $\Re$. Since we cannot add any first-order sentence γ to $\Gamma_{\Re}$ because the resulting set of sentences would be inconsistent (i.e., γ and $\neg\gamma$ would both be present), we cannot do anything within first-order logic to remove the nonstandard models. But the existence of nonstandard models is really a side effect; the real issue is that there are important things that cannot be expressed in first-order logic.*

The obvious question, then, is: why not use some higher-order logic? Our reasons for not doing so are, as we have said, (1) there are fundamental technical problems with higher-order logics, (2) first-order logic is well developed, and (3) higher-order logics are familiar to only a relatively few experts. But if we cannot formulate a workable modeling methodology these three reasons are irrelevant. Consequently, we have to show that we can get a good modeling methodology within first-order logic in spite of its limitations. That this is possible is a major result of this presentation.

In passing, we note that we will use the existence of nonstandard models to show that certain deductions are not possible. To show that $\sum \not\vdash \alpha$, we will describe a structure

[2]A technical weakness of higher-order logics is that "not all things that are true" can be deduced. However, this is not the point here. β is just not a true statement about all structures that satisfy $\Gamma_{\Re}$.

[3]The word "model" is used here in two ways. First, there is a model of a discrete-event system. This is a set of first-order sentences $\sum$. Second, there is the use of "model" in first-order logic. In that case, a model is the same thing as a structure. That is, it is a semantic interpretation of the first-order language. A nonstandard model of the real line is this latter kind of model.

which satisfies $\sum \cup \{\neg\alpha\}$. The first such use occurs in this section.

Given, then, that we want a first-order theory of time, what should it be? $\Gamma_{\Re}$ is inappropriate because we do not have a practical way to specify it. Instead, we choose to characterize time with the axioms, denoted here by $\sum_T$, for ordered real closed fields [6, page 41]. The language is $\{\leq, +, \cdot, 0, 1\}$, where 0 and 1 are constant symbols.

The field axioms are:

$$\begin{array}{rl}
(\forall x, y, z) & [x + (y + z) = (x + y) + z] \\
(\forall x) & [(x + 0 = x) \wedge (0 + x = x)] \\
(\forall x)(\exists y) & [(x + y = 0) \wedge (y + x = 0)] \\
(\forall x, y) & [x + y = y + x] \\
(\forall x) & [1x = x1 = x] \\
(\forall x, y, z) & [x(yz) = (xy)z] \\
(\forall x, y) & [xy = yx] \\
(\forall x, y, z) & [x(y + z) = (xy) + (xz)] \\
(\forall x, y) & [xy = 0 \rightarrow (x = 0 \vee y = 0)] \\
 & 0 \neq 1 \\
(\forall x) & [(x \neq 0) \rightarrow (\exists y)(yx = 1)]
\end{array}$$

The order axioms are

$$\begin{array}{rl}
(\forall x) & [x \leq x] \\
(\forall x, y, z) & [(x \leq y) \wedge (y \leq z) \rightarrow (x \leq z)] \\
(\forall x, y) & [(x \leq y) \wedge (y \leq x) \rightarrow (x = y)] \\
(\forall x, y, z) & [(x \leq y) \rightarrow (x + z \leq y + z)] \\
 & 0 < 1
\end{array}$$

In addition, there is the axiom

$$(\forall x)(\exists y) \left[(y^2 = x) \wedge (y^2 + x = 0)\right] \tag{3}$$

and the two infinite sets of axioms, for each $n \geq 0$

$$(\forall x_o, x_1, ..., x_n) \left[(x_o^2 + x_1^2 + \cdots + x_n^2 = 0) \rightarrow ((x_o = 0) \wedge ... \wedge (x_n = 0))\right] \tag{4}$$

and for each odd n ≥ 0

$$(\forall x_o, x_1, ..., x_n)(\exists y) \left[(x_n y^n + x_{n-1}\ y^{n-1} + ... + x_1 y + x_0 = 0) \vee (x_n = 0)\right] \tag{5}$$

The latter two are called axiom schema in that they each describe an infinite set of axioms.

The theory of $\sum_T$, Th($\sum_T$), is the set of all first-order sentences over the language $\{\leq, +, \cdot, 0, 1\}$ that can be deduced from $\sum_T$. This theory is complete, that is, if α is a first-order sentence over $\{\leq, +, \cdot, 0, 1\}$, then either $\alpha \in$ Th ($\sum_T$) or $\neg\alpha \notin$ Th ($\sum_T$) but not both. However, there are other first-order sentences that are true of the real line. For example, if we add a predicate symbol to describe exponentiation, y^x, there will be new first-order sentences that cannot be expressed in terms of the language $\{\leq, +, \cdot, 0, 1\}$. These sentences will be in $\Gamma_\Re$ but not in Th($\sum_T$). The point is that we are using part of the first-order theory of the real line: Th($\sum_T$) is a proper subset of $\Gamma_\Re$.

There are really two models recognized as standard models for $\sum_T$. The first, $(\Re, \leq, +, \cdot, 0, 1)$, is ordered real closed field in the real line. This is our intended model of time. The other is $(\mathcal{A}, \leq, +, \cdot, 0, 1)$, where $\mathcal{A}$ is the set of all algebraic numbers in the real line. The former is an uncountable model and the latter is countably infinite, so they are not isomorphic. We note in passing that the model $(\mathcal{A}, \leq, +, \cdot, 0, 1)$ plays a special role in the theory of ordered real closed fields in that it can be isomorphically embedded into any ordered real closed field. Though $(\Re, \leq, +, \cdot, 0, 1)$ is our intended model of time $(\mathcal{A}, \leq, +, \cdot, 0, 1)$ is sometimes referred to for technical reasons.

In any event, there is no first-order way to say we mean $(\Re, \leq, +, \cdot, 0, 1)$ and not $(\mathcal{A}, \leq, +, \cdot, 0, 1)$. Moreover, there are nonstandard models that are not isomorphic to either of these models.

If β is sentence (2) about least upper bounds, then any structure satisfying $\sum_T \cup \{\beta\}$ is isomorphic to $(\Re, \leq, +, \cdot, 0, 1)$. That is, if we add β, we get precisely the ordered real closed field in $\Re$. However, adding even just this one second-order sentence introduces the problems associated with higher-order logics, so we do not add it.

5. Σ_{PD}, Pseudo-differentiation

Consider a dynamic predicate family $\{P_-(\tau, x_1, ..., x_n), \delta P(\tau, x_1, ..., x_n)\}$. Our intention is that P_- should be left-piecewise constant, δP should be impulsive, and δP should be the pseudo-derivative of P_-. We define these terms in this section.

Definition 1 *A predicate $P_-(\tau, x_1, ..., x_n)$ is left-piecewise constant at time τ if the following first-order well-formed formula is satisfied by P_- at time τ.*

$$(\exists\tau_\ell)(\exists\tau_u)\,[[\tau_\ell < \tau < \tau_u] \wedge (\forall\tau_1)(\forall\tau_2)\,[((\tau_\ell < \tau_1, \tau_2 \leq \tau) \vee (\tau < \tau_1, \tau_2 < \tau_u)) \\ \rightarrow (\forall x_1, ..., x_n)\,\{P_-(\tau_1, x_1, ..., x_n) \leftrightarrow P_-(\tau_2, x_1, ..., x_n)\}]] \tag{6}$$

Remark 6 *Our convention is to use a "$-$" subscript to denote dynamic predicates and functions that are normally left-piecewise constant. Those that are right-piecewise constant will have a "+" subscript.*

Remark 7 *In a nonstandard model of $\sum_T$ it is possible that $(\tau_u - \tau_\ell)$ could be a so-called infinitesimal [5, page 164] even though it is greater than zero, that is, it could be smaller than each algebraic number. Or $(\tau_u - \tau_\ell)$ might be greater than each algebraic number, that is, it might be an infinite number. Either case is a technical curiosity because any first-order statements we deduce will be correct statements about all possible models of time, in particular, about the real line which is the one that we care about.*

Definition 2 *A predicate $\delta P(\tau, x_1, ..., x_n)$ is said to be impulsive at time τ if the following first-order well-formed formula is satisfied by δP at time τ.*

$$(\exists\tau_\ell)(\exists\tau_u)\left[[\tau_\ell < \tau < \tau_u] \wedge (\forall\tau')\left[\Big((\tau_\ell < \tau' < \tau) \vee (\tau < \tau' < \tau_u)\Big)\right.\right.$$
$$\left.\left.\rightarrow (\forall x_1, ..., x_n)(\neg\delta P(\tau', x_1, ..., x_n))\right]\right] \tag{7}$$

That is, δP is empty for time sufficiently close to τ, and it may or may not be empty at exactly τ. If it is nonempty, we say that there is an *impulse* at τ.

Remark 8 *Remark 7 about nonstandard models applies here also.*

Remark 9 *We will usually use a "δ" prefix as in "δP" to denote dynamic predicates that are normally impulsive.*

Rather than define the pseudo-derivative at one time, we have to define it in a neighborhood of a time; otherwise, we do not capture the intuitive idea of δP being the step change in P_.

Definition 3 *A predicate $\delta P(\tau, x_1, ..., x_n)$ is the left pseudo-derivative of a predicate $P\_(\tau, x_1, ..., x_n)$ in a neighborhood of a time τ if the following first-order well-formed formula is satisfied by δP and $P\_$ at time τ.*

$$(\exists\tau_\ell)(\exists\tau_u)[\tau_\ell < \tau < \tau_u] \wedge \left[(\forall\tau')(\tau_\ell < \tau' < \tau_u) \rightarrow \left[(\exists\tau_\ell')(\exists\tau_u')\left[\tau_\ell < \tau_\ell' < \tau' < \tau_u' < \tau_u\right]\right.\right.$$
$$\wedge \left[(\forall\tau_1)(\forall\tau_2)\Big((\tau_\ell' < \tau_1 \leq \tau') \wedge (\tau' < \tau_2 < \tau_u')\Big) \rightarrow (\forall x_1, ..., x_n)\Big\{\delta P(\tau', x_1, ..., x_n)\right.$$
$$\leftrightarrow (P\_(\tau_1, x_1, ..., x_n) + P\_(\tau_2, x_1, ..., x_n))\}]]] \tag{8}$$

where "+" denotes symmetric difference. τ_ℓ and τ_u establish an open neighborhood of τ in which δP is the pseudo-derivative of P_. In particular, for each τ' in this open neighborhood there is an open neighborhood of τ' defined by τ_ℓ' and τ_u'. Then for any τ_1 and τ_2 in this latter neighborhood with $\tau_1 \leq \tau'$ and $\tau' < \tau_2$ we have that $\delta P(\tau', x_1, ..., x_n)$ is the symmetric difference of $P\_(\tau_1, x_1, ..., x_n)$ and $P\_(\tau_2, x_1, ..., x_n)$.

The following lemma shows that the above definition captures the intuitive idea of pseudo-derivative.

Lemma 1 *If δP is the left pseudo-derivative of P_- in a neighborhood of time τ, then δP is impulsive and P_- is left-piecewise constant at time τ.*

Proof: Let $\tau' = \tau$. We know that there are τ_ℓ' and τ_u' such that $\delta P(\tau, x_1, ..., x_n) \leftrightarrow P_-(\tau_1, x_1, ..., x_n) + P_-(\tau_2, x_1, ..., x_n)$ for all $(\tau_\ell' < \tau_1 \leq \tau) \wedge (\tau < \tau_2 < \tau_u')$. From this we conclude that $P_-(\tau_3, x_1, ..., x_n)$ is constant for $\tau_\ell' < \tau_3 \leq \tau$ and for $\tau < \tau_3 < \tau_u'$ which shows that P_- is left-piecewise constant at time τ. Let τ_4 satisfy $\tau_\ell' < \tau_4 \leq \tau$. Since δP is the left pseudo-derivative at P_-, we have $\delta P(\tau_4, x_1, ..., x_n) \leftrightarrow P_-(\tau_5, x_1, ..., x_n) + P_-(\tau_6, x_1, ..., x_n)$ for all τ_5 and τ_6 is an open neighborhood of τ_4 contained in the interval $\left(\tau_\ell', \tau\right]$. Since it is constant in this neighborhood of τ_4, $\neg\delta P(\tau_4, x_1, ..., x_n)$ is satisfied. Therefore, δP is impulsive at time τ.

Henceforth, we will drop "left" from "left pseudo-derivative".

Remark 10 *Note that had we defined pseudo-derivative at a point rather than in a neighborhood, the above lemma would not be true.*

So far we have said that "δP is supposed to be the pseudo-derivative of P_-". We formalize this idea with a first-order sentence that says that if δP is impulsive at time τ, then it is the pseudo-derivative of P_- in a neighborhood of time τ. That is, for each dynamic family we require that the following sentence be satisfied.

$$(\forall\tau)\left[\{\text{Formula 7}\} \rightarrow \{\text{Formula 8}\}\right] \tag{9}$$

These sentences are part of $\sum_{PD}$. The rest of $\sum_{PD}$ is required for technical reasons. The additional sentences say that if δP is empty over any time interval, then P_- is constant over that interval. These sentences are

$$(\forall\tau_u)(\forall\tau_\ell)\left[\left\{(\forall\tau')\left((\tau_\ell < \tau' < \tau_u) \rightarrow (\forall x_1, ..., x_n)\left(\neg\delta P(\tau', x_1, ..., x_n)\right)\right)\right\}\right.$$
$$\left.\rightarrow \{(\forall\tau_1)(\forall\tau_2)\left((\tau_\ell < \tau_1, \tau_2 < \tau_u) \rightarrow (\forall x_1, ..., x_n)\left(P_-(\tau_1, x_1, ..., x_n) \leftrightarrow P_-(\tau_2, x_1, ..., x_n)\right)\right)\}\right] \tag{10}$$

$$(\forall\tau_\ell)\left[\left\{(\forall\tau')\left((\tau_\ell < \tau') \rightarrow (\forall x_1, ..., x_n)\left(\neg\delta P(\tau', x_1, ..., x_n)\right)\right)\right\}\right.$$
$$\left.\rightarrow \{(\forall\tau_1)(\forall\tau_2)\left((\tau_\ell < \tau_1, \tau_2) \rightarrow (\forall x_1, ..., x_n)\left(P_-(\tau_1, x_1, ..., x_n) \leftrightarrow P_-(\tau_2, x_1, ..., x_n)\right)\right)\}\right] \tag{11}$$

$$(\forall\tau_u)\left[\left\{(\forall\tau')\left((\tau' < \tau_u) \rightarrow (\forall x_1, ..., x_n)\left(\neg\delta P(\tau', x_1, ..., x_n)\right)\right)\right\}\right.$$
$$\left.\rightarrow \{(\forall\tau_1)(\forall\tau_2)\left((\tau_1, \tau_2, < \tau_u) \rightarrow (\forall x_1, ..., x_n)\left(P_-(\tau_1, x_1, ..., x_n) \leftrightarrow P_-(\tau_2, x_1, ..., x_n)\right)\right)\}\right] \tag{12}$$

Remark 11 *We need the second set of sentences in $\sum_{PD}$ because it is not implied by the first set. The reason is, once again, that there are nonstandard models of time.*

6. $\sum_{DEB}$, Discrete-event Behavior

A dynamic predicate δP has discrete-event behavior if it has a finite number of impulses within each finite time interval. In this case δP is impulsive at each time, and, therefore, from $\sum_{PD}$, it is the pseudo-derivative of P_- in a neighborhood of each time. Further, P_- will be left-piecewise constant with times of discontinuity the same as the impulse times of δP.

Remark 12 *There is no first-order formula which defines a finite time interval and none that defines a finite set, so we have to get at discrete-event behavior indirectly. We do so as follows.*

Definition 4 *A predicate family $\{\delta P,\ P_-\}$ is said to have discrete-event behavior if the following sentences are satisfied.*

$$(\forall \tau_\ell)(\forall \tau_u)\left[(\tau_\ell < \tau_u) \rightarrow (\exists \varepsilon)(0 < \varepsilon) \wedge (\forall \tau_1)(\forall \tau_2)\left\{(\tau_\ell < \tau_1, \tau_2 < \tau_u) \wedge (\tau_1 \neq \tau_2) \wedge \right.\right.$$
$$\left.\left.(\exists x_1, ..., x_n)\delta P(\tau_1, x_1, ..., x_n) \wedge (\exists x_1, ..., x_n)\delta P(\tau_2, x_1, ..., x_n) \rightarrow (\varepsilon < |\tau_1 - \tau_2|)\right\}\right] \tag{13}$$

$$(\forall \tau)\left[(\exists \tau_f)(\tau < \tau_f) \wedge (\exists x_1, ..., x_n)\delta P(\tau_f, x_1, ..., x_n) \rightarrow (\exists \tau_n)(\tau < \tau_n) \wedge (\exists x_1, ..., x_n)\delta P(\tau_n, x_1, ..., x_n)\right.$$
$$\left.\wedge (\forall \tau_b)\left\{(\tau < \tau_b < \tau_n) \rightarrow (\forall x_1, ..., x_n)\left(\neg \delta P(\tau_b, x_1, ..., x_n)\right)\right\}\right] \tag{14}$$

$$(\forall \tau)\left[(\exists \tau_f)(\tau_f < \tau) \wedge (\exists x_1, ..., x_n)\delta P(\tau_f, x_1, ..., x_n) \rightarrow (\exists \tau_p)(\tau_p < \tau) \wedge (\exists x_1, ..., x_n)\delta P(\tau_p, x_1, .., x_n)\right.$$
$$\left.\wedge (\forall \tau_e)\left\{(\tau_p < \tau_e < \tau) \rightarrow (\forall x_1, ..., x_n)\left(\neg \delta P(\tau_e, x_1, ..., x_n)\right)\right\}\right] \tag{15}$$

The above sentences for each dynamic family are the set $\sum_{DEB}$. The first sentence says that in each interval defined by τ_ℓ and τ_u, there is a lower bound ε on the spacing between impulses of δP. The second sentence says that if there is an impulse after time τ, then there is a next impulse after τ at time τ_n. The third sentence says the same thing about predecessors.

Remark 13 *We need all three sentences (13), (14), and (15); for example, $\sum_T \cup \{$first sentence$\}$ does not imply the second sentence. That is, there are structures satisfying $\sum_T \cup \{$first sentence$\}$ which do not satisfy the second. For example, let T be a nonstandard model of time with the real line $\Re$ embedded in it and with positive infinite times. Let δP be nonempty at 0 and empty everywhere else in $\Re$. Further, let there be a doubly infinite sequence $\{x_k\}$, k = ..., -1, 0, 1, ..., of positive infinite times (i.e. x_k is greater than every real number) such that $x_{k+1} = x_k + 1$ for all k. For times in $T - \Re$, let δP be nonempty at exactly these times. This δP will satisfy the first sentence but not the second. In particular, there is no next impulse after time 0 even though there are impulses after time 0.*

Remark 14 *A variation on the Definition (4) replaces the first sentence by one that treats only intervals of length one. This definition is implied by Definition (4) but not vice versa. However, this is another technical curiosity because they are equivalent when restricted to the real line.*

Remark 15 *Yet another technical curiosity is that for a nonstandard model of time a predicate δP can have discrete-event behavior with the bound ε being an infinitesimal, that is ε is greater than zero but less than every algebraic number. So we might have δP satisfying Definition (4) with an infinite number of pulses between 0 and 1. Again, we do not care about this possibility because every first-order sentence applied to the real line will have a standard and familiar meaning.*

7. $\sum_{RS}$, Pseudo-differential-difference Equations

Consider a simple ordinary differential-difference equation $dx/d\tau = f(x(\tau), x(\tau - 1))$. The right side f gives a function of time, the left side says that if f is the derivative of something, then one of them is x. We use this two part view as a model for our pseudo- differential-difference equations. $\sum_{PD}$ covers one part. A set of first-order sentences $\sum_{RS}$ — "RS" for "right sides" — covers the other part. For each endogenous dynamic predicate symbol family we have a sentence in $\sum_{RS}$ defining δP for times greater than or equal to the initial time τ_o.

Example 5 *Continuing the transport system example, the pseudo-differential-difference equations for the two endogenous families are*

$$(\forall\tau)\,[(\tau_o \le \tau) \rightarrow (\forall l)\,\{\delta W(\tau, l) \leftrightarrow ((\delta M(\tau) \wedge W_{-}(\tau, l)) + ((\tau = \tau_a) \wedge S_{-}(\tau, l, \tau_a)))\}] \tag{16}$$

$$(\forall\tau)\,[(\tau_o \le \tau) \rightarrow (\forall l)(\forall\tau_a)\,\{\delta S(\tau, l, \tau_a) \leftrightarrow \tag{17}$$

$$((\delta M(\tau) \wedge [W_{-}(\tau, PL(l)) \wedge (\tau_a = \tau + 1)]) + ((\tau = \tau_a) \wedge S_{-}(\tau, l, \tau_a)))\}] \tag{18}$$

that is, δW and δS have impulses when moves start or finish, and a move takes one second. PL is a function symbol denoting the "predecessor location" in the fixed loop. PL is an example of a static symbol. We refer to the subformula to the right of "$\leftrightarrow$" as the "right side".

Remark 16 *We will usually drop the outer $(\forall\tau)$ and the leading $(\tau_o \le \tau) \rightarrow$ and consider them implicit.*

The sentences in $\sum_{RS}$ are the heart of a model because they specify when events occur and what the events are. Right sides are usually causal in the sense that their current value is independent of the future. Our models allow "parallelism" in the sense that pseudo-differential-difference equations can easily model more than one event occurring at a time.

8. $\sum_S$, Static Quantities

These sentences define static quantities other than those defined in $\sum_T$. $\sum_S$ has no particular form; it depends on each particular case.

9. $\sum_H$, Histories

Histories are convenient for specifying initial conditions. The histories H₋ and δH with respect to a time τ_h of P₋ and δP, respectively, are defined by

$$(\forall\tau)(\forall x_1,...,x_n)(\forall\tau_h)\,[H\_(\tau,x_1,...,x_n,\tau_h) \leftrightarrow (\tau < \tau_h) \wedge P\_(\tau,x_1,...,x_n) \\ +(\tau_h \leq \tau) \wedge P\_(\tau_h,x_1,...,x_n)] \tag{19}$$

and

$$(\forall\tau)(\forall x_1,...,x_n)(\forall\tau_h)\,[\delta H(\tau,x_1,...,x_n,\tau_h) \leftrightarrow (\tau < \tau_h) \wedge \delta P(\tau,x_1,...,x_n)] \tag{20}$$

That is, the history of P₋ with respect to τ_h agrees with P₋ up to and including τ_h and is constant for times greater than or equal to τ_h. The history of δP with respect to τ_h agrees with δP for times less than τ_h and is empty for times greater than or equal to τ_h.

There is a pair of such sentences for each endogenous[4] dynamic family. The set of these sentences is denoted $\sum_H$. Each $\{H, \delta H\}$ is a new dynamic symbol family that is included in the language $\mathcal{L}$, and like all dynamic families it is required to satisfy $\sum_{PD}$.

10. $\sum_{IC\&Ex}$, Initial Conditions and Inputs

Initial conditions are conditions on histories with respect to the initial time τ_o. The set of inputs are, again, the exogeneous dynamic families. The set of sentences $\sum_{IC\&Ex}$ characterizes both. It always says that each H₋ is constant for $\tau_o \leq \tau$ and that each δH is empty for $\tau_o \leq \tau$. It also says that each family $\{H\_, \delta H\}$ has discrete-event behavior.

Often the needed initial conditions are conditions on the history H₋ at only $\tau = \tau_o$, that is, conditions on $H\_(\tau_o, x_1, ..., x_n, \tau_o)$. This is usually the case with pseudo-differential equations. In such cases we will additionally require that H₋ be constant for $\tau \leq \tau_o$ and δH be empty for $\tau \leq \tau_o$. We could have truncated time to $[\tau_o, \infty)$, but it is more convenient to have time be $\Re$ in all cases. So we accept the minor cost of having to be concerned with the histories for all times, even when most of these times are irrelevant as far as the solution of interest is concerned.

[4]We do not bother with histories of exogenous dynamic families in $\sum$ because they do not have initial conditions, at least not in $\sum$.

At one extreme each history and each input can be explicitly defined by a first-order formula that uses only static symbols. For example,

$$H\_(\tau, x, y, \tau_o) \leftrightarrow (x = c_1) \wedge (y = c_2) \tag{21}$$

where c_1 and c_2 are constant symbols in $\mathcal{L}$. Here H_ is constant for all time.

$$\delta H(\tau, x, y, \tau_o) \leftrightarrow \perp \tag{22}$$

that is, δH is empty for all times ($\perp$ is false), and

$$\delta I(\tau) \leftrightarrow (\tau = 1) \vee (\tau = 10) \vee (\tau = 32) \tag{23}$$

that is, there is an impulse at times 1, 10, and 32.

The problem with the above explicit way of characterizing initial conditions and inputs is that it is too limited. We need implicit characterizations. For example, a periodic input might be characterized.

$$\begin{array}{ll} \delta I(0), & (\forall\tau)\,[\delta I(\tau) \leftrightarrow \delta I(\tau+1)]\,, \\ \multicolumn{2}{l}{(\forall\tau)(\forall\tau_1)(\forall\tau_2)\,[(\tau < \tau_1, \tau_2 < \tau + 1) \wedge \delta I(\tau_1) \wedge \delta I(\tau_2) \rightarrow (\tau_1 = \tau_2)]} \end{array} \tag{24}$$

The first sentence says that there is an impulse at 0, the second says that δI is periodic, and the third says that any open interval of length 1 contains at most one impulse. This is an example of a δI that cannot be defined explicitly as we did above.

Remark 17 *Denote the union of $\sum_T$ and the sentences (24) by $\sum_{\delta I}$. We make a distinction between $\sum_{\delta I}$ being an implicit characterization of δI and $\sum_{\delta I}$ being an implicit definition of δI. The phrase "implicit characterization" is not formal, any set of sentences can be one. On the other hand, "implicit definition" has a precise meaning.*

Definition 5 *$\sum_{\delta I}$ defines δI implicitly [6, page 87] if*

$$\textstyle\sum_{\delta I}(\delta I_1) \cup \sum_{\delta I}(\delta I_2) \vdash (\forall\tau)\,[\delta I_1(\tau) \leftrightarrow \delta I_2(\tau)] \tag{25}$$

where δI_1 and δI_2 are new symbols, and $\sum_{\delta I}(\delta I_i)$ is $\sum_{\delta I}$ with each occurrence of δI replaced by δI_i, $i = 1,2$.

The above definition says that there is only one δI over a given model of time.

Beth's theorem says that $\sum_{\delta I}$ defines δI implicitly if and only if δI has an explicit definition. δI has an explicit first-order definition if there is a first-order formula ϕ with no occurrence of δI such that

$$(\forall\tau)\,[\delta I(\tau) \leftrightarrow \phi(\tau)] \tag{26}$$

is implied by $\sum_{\delta I}$.

However, it is not difficult to find two different δI*'s over the same model of time. For example, let* T *be a nonstandard model of time that contains* $\Re$ *as a proper subset and with infinitesimal and infinite times. Let* δI_1 *and* δI_2 *agree on* $\Re$ *but be shifted by 1/2 with respect to one another on the rest of* T*. Both satisfy* $\sum_{\delta I}$ *but are different. Therefore,* $\sum_{\delta I}$ *is not an implicit definition of* δI*. Consequently,* δI *cannot be defined explicitly. The point, then, is that* $\sum_{IC\&Ex}$ *need not be made up of explicit definitions of histories and inputs.*

Remark 18 *However, it is always possible that the first-order theory of the histories and inputs will be complete. That is,* $Th(\sum_B \cup \sum_{IC\&E_x})$ *is complete, where* $\sum_B$ *is the mathematical background needed to support* $\sum_{IC\&E_x}$*. Typically,* $\sum_B$ *contains* $\sum_T$ *and* $\sum_S$ *plus the parts of* $\sum_{PD}, \sum_{DEB},$ *and* $\sum_\omega$ *pertaining to historics and inputs. It also usually contains the part of* $\sum_{PD}$ *referring to historics and inputs.* $\sum_B$ *will also contain the part of* $\sum_\infty$ *referring to historics and inputs.*

The point is that $Th(\sum_B \cup \sum_{IC\&E_x})$ *can be complete even if there is not an implicit definition.*

Remark 19 *We refer to complete theories and completions of theories in a number of places. It should be appreciated that it is impractical to try to determine if an arbitrary set of sentences has a complete theory. Complete theories and completions have a mainly theoretical interest. In fact, even if we have not given a complete first-order description of the histories and inputs, it can easily be that what is missing is merely a technical curiosity. For example, the sentences (24) together with* $\sum_B$ *do not have a complete theory. Nevertheless, the standard model is uniquely defined (but not in first-order logic). The point is that we have enough information to characterize the things of practical importance.*

11. Solutions

A *solution* for given initial conditions and input is any standard structure that satisfies $\sum_t = \sum \cup \sum_{IC\&Ex} \cup \sum_H$. We also call a solution a *trajectory*. $\sum_H$ is needed to connect $\sum_{IC\&Ex}$ and $\sum$.

Remark 20 *"Standard structure" is not a technical phrase, and it is, of course, not definable within first-order logic. It does, however, say what our actual intention is.*

Remark 21 *Two structures are said to be elementarily equivalent [6, page 32] if they both satisfy exactly the same set of first-order sentences. Complete first-order theories and elementary equivalence classes of structures go together. Given a complete first-order theory,* *the class of all structures satisfying this theory is, by definition, an elementary equivalence class. Given an elementary equivalence class, each structure satisfies the same complete theory. The point is that technically we work with elementary equivalence classes or, equivalently, complete theories whether we say "standard structure" or not.*

12. Existence of a Solution

Most ordinary differential equations have a solution for a given input and set of initial conditions. Those that do not are usually of limited interest for systems applications. This is not the case for pseudo-differential-difference equations. A solution can easily fail to exist in an important way because we require that solutions have discrete-event behavior. This is illustrated in the next example.

Example 6 *The system being modeled has two entities which alternate between being in a predicate; as soon as one is in the predicate it is immediately replaced by the other one. Obviously, there will be an infinite number of alternations within any finite interval. In those cases, where neither or both entities are in the predicate, the system does nothing.*

The model has one endogenous family $\{C_-(\tau, x), \delta C(\tau, x)\}$ and two constant symbols "a" and "b". There is no exogeneous family. The initial time is 0. The only static condition, other than those in $\sum_T$, is a sentence saying that a and b are different. The pseudo-differential equation is

$$\delta C(\tau, x) \leftrightarrow [\{C_-(\tau, a) + C_-(\tau, b)\} \wedge ((x = a) \vee (x = b))] \tag{27}$$

That is, a and b alternate whenever just one of them is in C_-. There are two cases to consider, depending on the choice of initial conditions.

First, suppose that only one of the entities denoted by a and b is in C_- at time 0. In that case $\delta C(0,x)$ is not empty. This means that a and b are interchanged at time 0. If $\{C_-, \delta C\}$ has discrete-event behavior, there is supposed to be a time interval of nonzero length over which C_- is constant and δC is empty. But immediately after time 0, δC is again nonempty, which is a contradiction. In other words, for these initial conditions, $\sum \cup \sum_{IC\&Ex} \cup \sum_H$ is an inconsistent set of sentences. On the other hand, it is not difficult to construct a structure satisfying $\sum \cup \sum_{IC\&Ex} \cup \sum_H - \sum_{DEB}$. For example, let C_- contain a on the rational numbers and b on the irrational numbers. The resulting δC is not impulsive anywhere, so it does not have to be the pseudo-derivative of C_-. That is, if we drop $\sum_{DEB}$ we obtain a consistent set of sentences. The point is that for these initial conditions the system is attempting to alternate too fast to have discrete-event behavior.

Second, suppose that both a and b or neither is in C_- at time 0. In this case it is easy to see that there is a solution with C_- constant and δC empty for all time. Here, then,

$\sum \cup \sum_{IC\&Ex} \cup \sum_H$ is a consistent set of sentences. And, of course, $(\sum \cup \sum_{IC\&Ex} \cup \sum_H) - \sum_{DEB}$ is also consistent. Further, any structure with discrete-event behavior satisfying $\sum \cup \sum_{IC\&Ex} \cup \sum_H$ will satisfy $(\sum \cup \sum_{IC\&Ex} \cup \sum_H) - \sum_{DEB}$. However, there are also structures without discrete-event behavior satisfying this latter set of sentences and not the former. The point is that the existence of bizarre structures satisfying $(\sum \cup \sum_{IC\&Ex} \cup \sum_H) - \sum_{DEB}$ is physically significant only when $\sum \cup \sum_{IC\&Ex} \cup \sum_H$ is inconsistent.

From one point of view there is a solution if $\sum \cup \sum_{IC\&Ex} \cup \sum_H$ is a consistent set of sentences. However, we are not interested in *any* solution; we want "standard solutions". Each sort has a standard interpretation (e.g., the real line for time, the natural numbers for counting and arithmetic, etc.), and we are concerned with the existence of solutions based on these standard interpretations.

The usual way to proceed is to construct a structure that (1) satisfies $\sum \cup \sum_{IC\&Ex} \cup \sum_H$ and (2) uses the standard interpretations. That is, one shows the existence of a solution by constructing one. And, when doing so one may use any part of mathematics that helps with carrying out the construction, that is, we are not limited by first-order logic in our search for a structure satisfying $\sum \cup \sum_{IC\&Ex} \cup \sum_H$. Assuming that there are no inconsistencies in $(\sum \cup \sum_{IC\&Ex} \cup \sum_H) - \sum_{DEB}$, the construction will hinge on showing that the impulses in δC's will have to be separated from one another in time. Usually, this is not a major problem.

Remark 22 *The above argument is vague. Ideally, we would like to be able to say that there is a standard solution if and only if $\sum \cup \sum_{IC\&Ex} \cup \sum_H$ is consistent. However, the author does not have a way to guarantee that this is the case. However, even if the author had such a guarantee, showing consistency would still reduce to constructing a solution as sketched above.*

In summary, existence is not a crucial practical problem for ordinary differential equations, but it may be for our equations. Thus, the question cannot be completely ignored. We have to be assured that we have discrete-event behavior.

13. Explicit Solutions

As in the case of ordinary differential-difference equations, some, but definitely not all, pseudo-differential-difference equations have "closed form" or explicit solutions. Let

$$\delta P(\tau, ...) \leftrightarrow \phi(P_-, \delta I)(\tau, ...) \tag{28}$$

be some pseudo-differential equation, and let τ_o be the defined initial time. An explicit solution is a pair of first-order formulas $\psi_-(\tau, ...)$ and $\delta\psi(\tau, ...)$ with no occurrence of P$_-$ or

δP such that

$$\begin{array}{lcl}(\forall\tau)\,[(\tau \leq \tau_o) & \rightarrow & (\forall ...)\,\{\psi(\tau, ...) \leftrightarrow H\_(\tau, ..., \tau_o)\,\}] \\ (\forall\tau)\,[(\tau_o \leq \tau) & \rightarrow & (\forall ...)\,\{P\_(\tau, ...) \leftrightarrow \psi(\tau, ...)\,\}] \\ (\forall\tau)\,[(\tau < \tau_o) & \rightarrow & (\forall ...)\,\{\delta\psi(\tau, ...) \leftrightarrow \delta H(\tau, ..., \tau_o)\,\}] \\ (\forall\tau)\,[(\tau_o \leq \tau) & \rightarrow & (\forall ...)\,\{\delta P(\tau, ...) \leftrightarrow \delta\psi(\tau, ...)\,\}]\end{array}$$

can be deduced from $\sum_t$. Showing that the first and third sentences can be deduced is usually trivial because the histories of $\psi\_$ and $\delta\psi$ with respect to τ_o will always be defined to be H_$(\tau, ..., \tau_o)$ and $\delta H(\tau, ..., \tau_o)$. The challenge is to find the rest of $\psi\_$ and $\delta\psi$.

Example 7 *Consider*

$$\delta P(\tau, x) \leftrightarrow \delta I(\tau) \wedge (x = c) \wedge P\_(\tau, x) \tag{29}$$

where c is a constant symbol and τ_o = 0. In this case the following is the ψ part of an explicit solution

$$\begin{array}{l}\psi\_(\tau, x) \leftrightarrow [(\tau \leq 0) + \{(0 < \tau) \wedge [(x \neq c) + (x = c) \wedge (\forall\tau')\,((0 \leq \tau' < \tau) \rightarrow \neg\delta I(\tau'))]\,\}] \\ \qquad\qquad\qquad \wedge\ H\_(\tau, x, 0)\end{array} \tag{30}$$

where, recall, $H\_(\tau, x, 0)$ is constant for $0 \leq \tau$. Thus, for $\tau \leq 0$ $\psi\_$ is the history, and for $(0 < \tau)$ $\psi\_$ is constant until the first impulse of δI, if there is one. Then the element denoted by c is deleted from the value of $H\_$ if it is present; otherwise, nothing happens. Subsequent impulses have no effect. If the first impulse occurs at 0, then c, if present, is deleted at time 0.

Remark 23 *Having an explicit solution is a fairly strong property. It says that for every structure that satisfies $\sum_t$ the solution is represented by the formulas $\{\psi\_, \delta\psi\}$. In particular, this is the case even when nonstandard models of time are used. In any event, there are many applicable $\sum_t$'s that do not have an explicit solution.*

Remark 24 *If we have an explicit solution, then the solution is, in an important way, unique. Indeed, suppose the $\{P_{1\_}, \delta P_1\}$ and $\{P_{2\_}, \delta P_2\}$ are two solutions with the same histories and inputs. Then both satisfy the above four sentences and are, consequently, equal because $\{\psi\_, \delta\psi\}$ and $\{H\_, \delta H\}$ do not involve $\{P_1, \delta P_2\}$ and $\{P_{2\_}, \delta P_2\}$*

Remark 25 *Since $\delta\psi$ can be determined from ψ, we will often call ψ by itself the explicit solution.*

14. Unique Solutions

In the case of ordinary differential equations initial conditions and inputs usually determine a unique trajectory (solution). This is essentially the case with pseudo-differential-difference equations, and from a practical point of view uniqueness, or lack thereof, is usually obvious. For example, if $\sum$ contains a system of n pseudo-differential equations, and if $\sum_{IC\&Ex}$ defines all histories and inputs explicitly, then $\sum_t$ will almost always have a unique solution. However, this is informal, and if one needs to go further, there are several technical issues.

Remark 26 *There are four reasonable ways to define uniqueness.*

1. *There is a unique standard structure (up to isomorphism) satisfying $\sum_t$.*
2. *$\sum_t$ has an explicit solution.*
3. *$\sum_t$ is an implicit definition of a solution.*
4. *The theory of $\sum_t$ is complete, where the theory of $\sum_t$, denoted $Th(\sum_t)$, is the set of all first-order sentences that can be deduced from $\sum_t$.*

The first definition is the one that we really care about, but it cannot be expressed in first-order logic. The others are, in contrast, first-order conditions.

The second has already been discussed in section 13. The point made there is that many useful $\sum_t$'s do not have an explicit solution; consequently, the second definition would be too strong.

Consider the third definition. To simplify things, assume for a moment that $\sum_t$ has just one endogenous dynamic family $\{P_-, \delta P\}$. Let $\{P_{1-}, \delta P_1\}$ and $\{P_{2-}, \delta P_2\}$ be two dynamic families of the same type as $\{P_-, \delta P\}$. Let $\sum_t(P_{1-})$ and $\sum_t(P_{2-})$ denote $\sum_t$ with $\{P_-, \delta P\}$ replaced by $\{P_{1-}, \delta P_1\}$ and $\{P_{2-}, \delta P_s\}$, respectively. An obvious definition of uniqueness is

$$\textstyle\sum_t(P_{1-}) \cup \sum_t(P_{2-}) \vdash (\forall\tau)(\forall...)\,[P_{1-}(\tau,...) \leftrightarrow P_{2-}(\tau,...)] \wedge (\forall\tau)(\forall...)\,[\delta P_1(\tau,...) \leftrightarrow \delta P_2(\tau,...)] \tag{31}$$

that is, when any two trajectories have the same histories, inputs, and static quantities, then they are the same. $\sum_t$ is said to be an implicit definition of the solution if the deduction 31 is possible. This concept of uniqueness is essentially the one used for ordinary differential equations. Unfortunately, it is too strong. In fact, Beth's theorem says that definitions 2 and 3 are equivalent.

The fourth definition is the first-order definition which is generally applicable, but there are problems. First, suppose that $\sum_T \cup \sum_{PD} \cup \sum_{DEB} \cup \sum_S \cup \sum_{IC\&Ex}$ has a complete theory. This says that we have a complete first-order definition of histories, inputs, and static quantities. $\sum_T \cup \sum_{PD} \cup \sum_{DEB} \cup \sum_S$ are included to provide the mathematical foundation for $\sum_{IC\&Ex}$. If, then, $\sum_t$ is an implicit definition of the solution, then the theory of $\sum_t$ will be complete. However, as we had said, $\sum_t$ need not be an implicit definition; moreover, the theory of $\sum_t$ need not be complete even when the first definition is satisfied.

Suppose that $Th(\sum_t)$ is not complete but that $Th(\sum_T \cup \sum_{PD} \cup \sum_{DEB} \cup \sum_S \cup \sum_{IC\&Ex})$ is complete. Further, suppose that there is a unique standard structure satisfying the latter theory, that is, $\sum_{IC\&Ex}$ uniquely describes some real histories and inputs. Assuming that $\sum_t$ is consistent, as we usually do, $Th(\sum_t)$ will have a completion. This means that we can add sentences to obtain a complete theory, in fact there will be many possible completions. So to get a unique solution according to definition 4 we have to select one of these completions, that is, we have to add more first- order sentences to our model. The question is: which ones?

In a sense, the selection is easy. Suppose that definition 1 is satisfied and let S be the standard structure. The theory of S, denoted $Th(S)$, is the set of all first-order sentences satisfied by S. $Th(S)$ is always complete, and since S satisfies $\sum_t$, we have $Th(\sum_t) \subset Th(S)$. Since we have assumed $Th(\sum_t)$ not complete, this is a proper containment. The point is that the natural completion of $Th(\sum_t)$ is $Th(S)$, so it is the one to pick from all the possible completions of $Th(\sum_t)$.

Ideally, we would like to find a set of sentences $\sum_{add}$ to add to $\sum$ so that whenever $Th(\sum_T \cup \sum_{PD} \cup \sum_{DEB} \cup \sum_S \cup \sum_{IC\&Ex})$ was complete, $Th(\sum_t \cup \sum_{add})$ would be complete and have a unique standard structure satisfying it. That is, $\sum_{add}$ is in a sense a missing part of the model $\sum$. We discuss this in the next section.

15. $\sum_\omega$, Universal Quantification Over Time

If some formula $\psi(\tau)$ is satisfied over each interval $(-\infty, \tau_1]$, where τ_1 is finite, then we conclude that $(\forall z)\psi(\tau)$ is satisfied.

Example 8 *Consider the pseudo-differential equation*

$$\delta C(\tau, x) \leftrightarrow \delta I_a(\tau) \wedge (x = a) + \delta I_b(\tau) \wedge (x = b) \tag{32}$$

where a and b are constant symbols. The initial conditions are $(\forall\tau)\,[(\tau \leq 0) \rightarrow \neg H_-(\tau, c, 0)]$ and $(\forall\tau)\,[(\tau < 0) \rightarrow (\forall x)\,(\neg\delta H(\tau, x, 0))]$, where c is another constant. There are sentences in $\sum_S$ which say that a, b, and c are distinct and that any x is one of them. The input δI_a causes a to be added to C_- if it is absent and to be deleted if it is present. The analogous statement holds for b.

The initial conditions say that c is not initially in C_-. Since the pseudo-differential equation does not change the status of c, can we conclude that c is never in C_-. The answer is no. At least it is no, if we we do not include the set of sentences $\sum_\omega$ being discussed in this section. To see why, suppose that $\sum_\omega$ is not present.

Then let time T be a nonstandard model with infinite times that contains $\Re$ as a subset. It is easy to construct a solution such that C_- does not contain c on $\Re$ but does *contain c for some infinite times. Intuitively, T has more than one part, one of them is $\Re$, the initial conditions determine the solution on $\Re$, but not on the other parts.*

The point is that having c in C_- does not make practical sense, so we want to remove this possibility. That is the purpose of $\sum_\omega$.

Remark 27 *Since there is no first-order way to define a finite number, the statement opening the section must be modified to fit into first-order logic. There are several ways to do this, and we allow any one of them. For example, if we know that $\psi(\tau)$ is satisfied over every interval of the form $(\infty, a]$, where a is an algebraic number, we then conclude that $(\forall\tau)\psi(\tau)$ is satisfied. Strictly speaking, this introduces an infinite deduction rule because the deduction of $(\forall\tau)\psi(\tau)$ requires an infinite number of conditions — one for each algebraic number — therefore this is not really a first-order solution. Another approach is to show (1) that $\psi(\tau_o)$ is satisfied and (2) that $\psi(\tau) \rightarrow \psi(\tau) + \delta\psi(\tau)$ is satisfied for all τ. This might seem like begging the question because we are again confronted with universal quantification over time. However, in most applications either $(\forall\tau)\,[\psi(\tau) \rightarrow \psi(\tau) + \delta\psi(\tau)]$ or its negation is implied by $\sum \cup \sum_{IC\&Ex} \cup \sum_H$. The reason is that $\sum \cup \sum_{IC\&Ex} \cup \sum_H$ is usually a first-order characterization of what is going to happen over the next instant of time.*

Yet another way to proceed would be to (1) show that $\psi(\tau_o)$ is satisfied and then show that if $\psi(\tau)$ is satisfied at an arbitrary time τ, this fact implies that $\psi$is satisfied immediately after the next event, if there is one. From these facts we would again conclude $(\forall\tau)\psi(\tau)$. There are other imaginable approaches. We allow any one of them. All of them together are the set $\sum_\omega$ which is a subset of $\sum_{add}$ discussed in remark 26. In fact, it is usually the only part of $\sum_{add}$, that we make explicit.

Remark 28 ***The problem arises only if the theory of $\sum \cup \sum_{IC\&Ex} \cup \sum_H$ is not complete. Indeed, if this theory is complete, then for any $(\forall\tau)\psi(\tau)$ either it or its negation, but not both, must be in theory. However, if the theory $\sum \cup \sum_{IC\&Ex} \cup \sum_H$ is not complete and neither $(\forall\tau)\psi(\tau)$ nor $\neg(\forall\tau)\psi(\tau)$ is in this theory, then there is a completion containing $\sum \cup \sum_{IC\&Ex} \cup \sum_H \cup \{(\forall\tau)\psi(\tau)\}$ and another completion containing $\sum \cup \sum_{IC\&Ex} \cup \sum_H \cup \{\neg(\forall\tau)\psi(\tau)\}$. However, only one of them makes practical sense. It is the one that agrees with Th(S), where S, again, is the standard structure satisfying $\sum \cup \sum_{IC\&Ex} \cup \sum_H$. The various techniques for deciding whether $(\forall\tau)\psi(\tau)$ is satisfied are really techniques based on things that are true in the standard structure. That is, the sentences in $\sum_\omega$ will allow us to proceed without knowing S explicitly.***

Remark 29 ***The last two remarks beat about the bush because the author does not know of a better way to select the "natural" completion. This has no practical fall out, but it would be nice to tie things up better.***

Remark 30 ***There is a subtle trap to be avoided. It is connected with Beth's theorem. Suppose that $\sum_t(P_1)$ and $\sum_t(P_2)$ are two version of $\sum_t$ with two different sets of endogenous***

dynamic families substituted for the original endogenous dynamic families. If we let $\psi(\tau) = (\forall ...) [P_{1\_}(\tau, ...) \leftrightarrow P_{2\_}(\tau, ...)]$, *it will usually be the case that we can show that*

$$\psi(\tau_o) \wedge (\forall \tau) [\psi(\tau) \rightarrow \psi(P_+)(\tau)] \tag{33}$$

where $\psi(P_+)$ *is* ψ *with* $P_\_$ *replaced by* P_+, *is satisfied, so we conclude that* $(\forall \tau)\psi(\tau)$ *is always satisfied. One might then conclude — erroneously — that by adding a sentence to* $\sum_\tau$ *we have made it an implicit definition of the solution; therefore, we have an explicit definition of a solution. This is erroneous because we have not added a sentence about* $P_\_$ *to* $\sum$, *we have added a sentence about* $P_{1\_}$ *and* $P_{2\_}$ *to* $\sum(P_1) \cup \sum(P_2)$, *and that is not what Beth's theorem is about. In other words, we do not magically force an explicit solution into existence.*

Remark 31 *If more than one time variable is present or if the natural numbers are used, the sentences in* $\sum_\omega$ *will have to be augmented to handle these cases.*

Remark 32 *Something analogous to* $\sum_\omega$ *is built into most temporal logics [8, 9, 10, 11] in one way or another.*

16. Supervising

We present a brief overview of using our modeling methodology in the supervision of discrete-event systems. By "supervision" we mean constraining the inputs to a discrete-event system in such a way that some condition is satisfied. The problem is usually that some inputs cannot be constrained. Such inputs are said to be *uncontrollable*; the others are *controllable* inputs.

To simplify the discussion we assume for a moment that our first-order language has one endogeneous dynamic family $\{P_\_, \delta P\}$ and two exogeneous dynamic families $\{u_\_, \delta u\}$ and $\{c_\_, \delta c\}$, where the first is uncontrollable and the second is controllable. Since we will never use them, we suppress $u_\_$ and $c_\_$. We assume discrete-event behavior for all dynamic entities.

Assume that we want a sentence $\psi(P_\_, \delta P, \delta u, \delta c)$ to be satisfied because it characterizes some kind of desirable behavior. From a practical point of view the set of standard structures satisfying ψ is the set of acceptable trajectories. This set can be thought of as a generalization of the event languages used with finite-state automaton models of discrete-event systems.

The examples presented in this section are relatively simple. However, it often happens that even in big discrete-event systems that the condition ψ will refer explicitly to only a small part of the system. This means that, with luck, one has a chance of attacking a supervision problem even in an enormous system. In any event, some of the examples in this section have infinite state sets, and, yet, the solution of the supervision problem is trivial.

Example 9 *Suppose that the sentence ψ is*

$$\psi = (\forall\tau)(\forall\tau_1)\,[P_{1\_}(\tau,\tau_1,a_3) \rightarrow (\tau \leq \tau_1 < \tau+1)] = (\forall\tau)\phi_{\_}(\tau) \tag{34}$$

where a_3 is a constant symbol; that is, the times τ_1 recorded in $P_{\_}$ are always in the interval $[\tau,\tau+1)$.

Now apply ψ to the discrete-event system characterized by

$$\delta P_1(\tau,\tau_1,x) \leftrightarrow (\tau=\tau_1)\wedge(x=a_3)\wedge P_{1\_}(\tau,\tau_1,a_3) + \delta c(\tau)\wedge(x=a_3)\wedge(\tau_1=\tau+1) \tag{35}$$

When there is an impulse in δc, the subformula $\delta c(\tau)\wedge(x=a_3)\wedge(\tau_1=\tau+1)$ adds the ordered-pair $<\tau+1,a_3>$ to $P_{1\_}$ if it is not already present and deletes if otherwise. The subformula $(\tau=\tau_1)\wedge(x=a_3)\wedge P_{1\_}(\tau,\tau_1,a_3)$ deletes an ordered-pair $<\tau_1,a_3>$ from $P_{1\_}$ when $\tau=\tau_1$. So if δc adds $<\tau+1,a_3>$ to $P_{1\_}$, it is automatically removed one second later.

Let 0 be the initial time and assume $\phi_{\_}(0)$ is satisfied. Since δu does not occur in this example, we will assume that δc is uncontrollable here.

Since we are assuming discrete-event behavior, $\phi_{\_}(\tau)$ has a pseudo- derivative given by

$$\begin{aligned}\delta\phi(\tau) \;&\leftrightarrow\; (\forall\tau_1)\,[P_{1\_}(\tau,\tau_1,a_3)\rightarrow(\tau\leq\tau_1<\tau+1)]\\ &+\; (\forall\tau')\left[\left(P_{1\_}(\tau,\tau_1',a_3)+\delta P_1(\tau,\tau_1',a_3)\right)\rightarrow(\tau<\tau_1'\leq\tau+1)\right]\end{aligned} \tag{36}$$

that is, $\phi_{\_}(\tau)+\phi_{+}(\tau)$, where, informally, $\phi_{+}(\tau)$ is $\phi_{\_}(\tau+)$, that is, $\phi_{\_}$ at a slightly later time.

Assuming that $\phi_{\_}(\tau)$ is satisfied, we want the second subformula on the right of sentence (36) to be satisfied. Substitution of δP_1 yields

$$(\forall\tau_1')\left[\left((\tau\neq\tau_1')\wedge P_{1\_}(\tau,\tau_1',a_3)+\delta c(\tau)\wedge(\tau_1'=\tau+1)\right)\rightarrow(\tau<\tau_1'\leq\tau+1)\right] \tag{37}$$

for this subformula, and it is implied by $\phi_{\_}(\tau)$. Indeed, $<\tau,\tau_1'>$ satisfies the subformula to the left of "$\rightarrow$" by either satisfying $(\tau\neq\tau_1)\wedge P_{1\_}(\tau,\tau_1',a_3)$ or with $\tau_1'=\tau+1$, and in either case $(\tau<\tau_1'\leq\tau+1)$ is satisfied. Thus, for any δc the sentence ψ is satisfied.

This is a trivial analysis even though this is an infinite state system. Trying to model this situation with a finite-state model could easily result in a very large state set and lengthy analysis.

Example 10 *Consider the following sentence*

$$\psi = (\forall\tau)\,[(\forall\tau_1)\,\{P_{1\_}(\tau,\tau_1,a_3) \rightarrow (\exists\tau_2)(\tau_1 \leq \tau_2) \wedge P_{2\_}(\tau,\tau_2,b_2)\}] = (\forall\tau)\phi_{\_}(\tau) \tag{38}$$

that is, any time there is a pair $< \tau_1, a_3 >$ in the value of $P_{1\_}$ there must be a pair $< \tau_2, b_2 >$ in the value of $P_{2\_}$ with $\tau_1 \leq \tau_2$.

Assuming, as in the previous example, discrete-event behavior, the pseudo-derivative of $\phi_{\_}(\tau)$ is

$$\begin{aligned}&\delta\phi(\tau) \leftrightarrow \phi_{\_}(\tau) + \\ &(\forall\tau_1)\,\{(P_{1\_}(\tau,\tau_1,a_3) + \delta P_1(\tau,\tau_1,a_3)) \rightarrow (\exists\tau_2)(\tau_1 \leq \tau_2) \wedge (P_{2\_}(\tau,\tau_2,b_2) + \delta P_2(\tau,\tau_2,b_2))\}\end{aligned}$$

Now suppose that we apply the above condition to the discrete-event system characterized by

$$\begin{aligned}\delta P_1(\tau,\tau_1,x) &\leftrightarrow (\tau = \tau_1) \wedge (x = a_3) \wedge P_{1\_}(\tau,\tau_1,a_3) + \delta c(\tau) \wedge (x = a_3) \wedge (\tau_1 = \tau + 1)\\ \delta P_2(\tau,\tau_2,y) &\leftrightarrow (\tau = \tau_2) \wedge (y = b_2) \wedge P_{2\_}(\tau,\tau_2,b_2) + \delta u(\tau) \wedge (y = b_2) \wedge (\tau_2 = \tau + 2)\end{aligned}$$

where δc is controllable and δu is uncontrollable, a_3 and b_2 are constant symbols, and the initial time is 0.

Substituting the definitions of δP_1 and δP_2 into $\delta\phi$ yields

$$\begin{aligned}\delta\phi(\tau) &\leftrightarrow \phi(\tau)\\ &+ \;\; [(\forall\tau_1)\,\{(\tau \neq \tau_1) \wedge P_{1\_}(\tau,\tau_1,a_3) + \delta c(\tau) \wedge (\tau_1 = \tau + 1)\}\\ &\rightarrow \;\; (\exists\tau_2)\,\{(\tau_1 \leq \tau_2) \wedge ((\tau \neq \tau_2) \wedge P_{2\_}(\tau,\tau_2,b_2) + \delta u(\tau) \wedge (\tau_2 = \tau + 2))\}]\end{aligned}$$

Since $\phi_{\_}$ is satisfied, $\delta\phi(\tau) \leftrightarrow \perp$ corresponds to the second subformula on the right being satisfied.

We also assume (see the previous example) that

$$(\forall\tau)(\forall\tau_1)\,[P_{1\_}(\tau,\tau_1,a_3) \rightarrow (\tau \leq \tau_1 < \tau + 1)] \tag{39}$$

and

$$(\forall\tau)(\forall\tau_2)\,[P_{2\_}(\tau,\tau_2,b_2) \rightarrow (\tau \leq \tau_2 < \tau + 2)] \tag{40}$$

are satisfied.

It follows from (39) that

$$(\tau_1 \neq \tau) \wedge P_{1\_}(\tau,\tau_1,a_3) \text{ and } \delta c(\tau) \wedge (\tau_1 = \tau + 1) \tag{41}$$

are mutually exclusive.
Likewise

$$(\tau_1 \leq \tau_2) \wedge (\tau \neq \tau_2) \wedge P_{2\_}(\tau, \tau_2, b_2) \text{ and } (\tau, \leq \tau_2) \wedge \delta u(\tau) \wedge (\tau_2 = \tau + 2) \tag{42}$$

are mutually exclusive because of (40). If $(\tau_1 \neq \tau) \wedge P_{1\_}(\tau, \tau_1, a_3)$ *is satisfied, then by* $\phi_{\_}(\tau)$ *and (39) it follows that* $(\exists \tau_2)(\tau_1 \leq \tau_2) \wedge (\tau \neq \tau_2) \wedge P_{2\_}(\tau, \tau_2, b_2)$ *is satisfied. If* $\delta c(\tau) \wedge (\tau_1 = \tau + 1)$ *is satisfied, then either* $(\exists \tau_2)(\tau_1 \leq \tau_2) \wedge (\tau \neq \tau_2) \wedge P_{2\_}(\tau, \tau_2, b_2)$ *or* $(\exists \tau_2)((\tau_1 \leq \tau_2) \wedge \delta u(\tau) \wedge (\tau_2 = \tau + 2))$ *have to be satisfied. After further simplification, we get*

$$\delta c(\tau) \rightarrow \delta u(\tau) \vee (\exists \tau_2)((\tau + 1 \leq \tau_2) \wedge P_{2\_}(\tau, \tau_2, b_2)) \tag{43}$$

that is, there can be an impulse in δc *if there is one in* δu *or if there is a pair* $< \tau_2, b_2 >$ *already in* $P_{2\_}$ *with* $\tau + 1 \leq \tau_2$. *This assumes that* δc *can depend on* δu. *If this is not possible, then we have*

$$\delta c(\tau) \rightarrow (\exists \tau_2)((\tau + 1 \leq \tau_2) \wedge P_{2\_}(\tau, \tau_2, b_2)) \tag{44}$$

That is, $\delta c(\tau)$ *cannot depend on* $\delta u(\tau)$ *because, for example, there is a significant delay between sensing the existence of an impulse* $\delta u(\tau)$ *and creating an impulse* $\delta c(\tau)$.

Example 11 *This is an extension of the preceding example.*

Suppose that in addition to satisfying ψ, *we want to select* δc *so that* $P_{1\_}(10, 10.5, a_3)$ *is satisfied. That is, we want to have a pair* $< 10.5, a_3 >$ *in the value of* $P_{1\_}$ *at time* $\tau =$ *10.*

Assuming discrete-event behavior, this is equivalent to

$$\delta P_{1\_}(9.5, 10.5, a_3) \wedge \neg P_{1\_}(9.5, 10.5, a_3) \tag{45}$$

being satisfied. But by (39) $\neg P_{1\_}(9.5, 10.5, a_3)$ *is satisfied. By the definition of* δP_1,

$$\delta P_1(9.5, 10.5, a_3) \leftrightarrow \delta c(9.5) \tag{46}$$

But from the preceding example

$$\delta c(9.5) \rightarrow \delta u(9.5) \vee (\exists \tau_2)(10.5 \leq \tau_2) \wedge P_{2\_}(9.5, \tau_2, b_2) \tag{47}$$

Since δu *is arbitrary,* $\delta u(9.5)$ *need not be satisfied, so we need* $(\exists \tau_2)(10.5 \leq \tau_2) \wedge P_{2\_}(9.5, \tau_2, b_2)$

to be satisfied. But if there is no appropriate impulse in δu before 9.5, the value of $P_{2\_}$ will be empty at $\tau = 9.5$. Thus the sentence

$$\psi \wedge P_{1\_}(10, 10.5, a_3) \tag{48}$$

is not controllable.

In general, we want to be able to select δc in such a way that ψ is satisfied for any δu. This means that allowable $\delta c's$ will be determined by δu. In fact, δc has to be causally related to δu.

Remark 33 *The foregoing approach can be formalized; however, the formalization requires higher- order logic. It is useful for stating the above condition precisely, but it is of very limited value for computational purposes. For example, one part of a formalization might be*

$$\begin{aligned}&\textstyle\sum_T \cup \sum_S \cup \sum_\Omega \vdash (\forall \delta u)\,[(\delta u\ DEB) \rightarrow \\ &(\exists \delta c)(\exists\,\{P_\_, \delta P\})\,\{\psi_\_(P_\_, \delta P, \delta u, \delta c) \wedge \textstyle\sum_{PD} \wedge \sum_{DEB}\}]\end{aligned} \tag{49}$$

$\sum_T$ and $\sum_S$ are, as before, the definitions of time and static quantities. $\sum_\Omega$ is a modification of $\sum_\omega$. In particular, if $\alpha(A,B,C)$ is a sentence in $\sum_\omega$ where A, B, and C are predicate symbols, then α is replaced by $(\forall X)(\forall Y)(\forall Z)\alpha(X,Y,Z)$ where X, Y, and Z are predicate variable symbols. The idea is that the first-order sentence $\alpha(A,B,C)$ which is a condition on specific predicates is replaced by the general condition $(\forall X)(\forall Y)(\forall Z)\alpha(X,Y,Z)$ on all predicates. It is reasonable to do this because $\sum_\omega$ is already a set of general statements about behavior as $\tau \rightarrow \infty$. The infinite set $\sum_T \cup \sum_S \cup \sum_\Omega$, in effect, sets up the mathematical environment. The deduction says that in this environment the second-order sentence on the right is satisfied. This sentence says that for every uncontrolled input δu with discrete-event behavior, $(\forall \delta u)\,[(\delta u DEB) \rightarrow$, there exists a controlled input δc and a dynamic family $\{P_\_, \delta P\}$ with discrete-event behavior and with δP the pseudo-derivative of $P_\_$ such that ψ is satisfied, $(\exists \delta c)(\exists\,\{P_\_, \delta P\})\,\{\psi(P_\_, \delta P, \delta u, \delta c)\} \wedge \sum_{PD} \wedge \sum_{DEB}$. It is a sentence because $\sum_{PD}$ and $\sum_{DEB}$ are finite and their conjunctions can be formed. It is second-order because we are treating $\delta u, \delta c, P_\_$, and δP as predicate variable symbols and quantifying over them.

The above formal condition says that there is a structure satisfying ψ for every δu. It does not say, however, that this structure is a trajectory of the discrete-event system satisfying the initial conditions or any further conditions on the inputs. To include the discrete-event system we add finite sets $\sum_{RS}$ and $\sum_{IC\&Ex}$ to the sentence on the right of " $\vdash$". The result is

$$\begin{aligned}&\textstyle\sum_T \cup \sum_S \cup \sum_\Omega \vdash (\ \forall \delta u)\,[(\delta u DEB) \rightarrow \\ &(\exists \delta c)(\exists\{P_\_, \delta P\})\,\{\psi(P_\_, \delta P, \delta u, \delta c) \wedge \textstyle\sum_{PD} \wedge \sum_{DEB} \wedge \sum_{RS} \wedge \sum_{IC\&Ex}\}]\end{aligned} \tag{50}$$

where, recall, $\sum_{RS}$ is the set of right sides for the pseudo-differential-difference equations.

However, the above condition does not say that δc is causally related to δu. This idea is captured by the following deduction.

$$
\begin{gathered}
\sum_T \cup \sum_S \cup \sum_\Omega \vdash \psi(P_-, \delta P, \delta u, \delta c) \wedge \sum_{RS} \wedge \sum_{IC\&Ex} \\
(\forall \tau) \Big[(\forall \delta u')(\forall \theta) \Big\{ (\theta < \tau) \rightarrow (\delta u(\theta) \leftrightarrow \delta u'(\theta)) \Big\} \\
\rightarrow \\
(\exists \{P_- \delta P\})(\exists \delta c')(\forall \theta) \Big\{ \Big((P_-(\theta) \leftrightarrow P'_-(\theta)) \Big) \wedge \\
\Big((\theta < \tau) \rightarrow \Big((\delta P(\theta) \leftrightarrow \delta P'(\theta)) \Big) \wedge \Big(\delta c(\theta) \leftrightarrow \delta c'(\theta) \Big) \Big) \Big\} \\
\wedge \sum_{PD} \wedge \sum'_{PD} \wedge \sum_{DEB} \wedge \sum'_{DEB} \wedge \sum_{RS} \wedge \sum'_{RS} \wedge \sum_{IC\&Ex} \wedge \sum'_{IC\&Ex} \wedge \psi \left(P', \delta P', \delta u', \delta c' \right) \Big]
\end{gathered} \tag{51}
$$

that is, whenever we have a trajectory for the system satisfying ψ and for any time τ, if the $\delta u'$ is a new uncontrolled input that agrees with the old one δu in the past, then we can always find a new trajectory using $\delta u'$ that agrees with the old one in the past and still satisfies the condition ψ. $\sum'_{PD}$ is $\sum_{PD}$ applied to the new trajectory, and so forth.

Example 12 *Let $\mathcal{L} = \{\leq, +, \cdot, 0, 1\} \cup \{P_{1-}, \delta P_1, P_{2-}, \delta P_2, \delta u, \delta c, a_1, a_2, a_3, b_1, b_2\}$, where a_1, a_2, a_3, b_1, b_2 are constant symbols denoting different entities. Let $\psi(\tau) = (\forall \tau)\phi_-(\tau)$, and $\phi_-(\tau) = \theta_{1-}(\tau) \vee \theta_{2-}(\tau)$, where $\theta_{1-}(\tau) = (\forall \tau_1)(\neg P_{1-}(\tau, \tau_1, a_3))$ and $\theta_{2-}(\tau) = (\forall \tau_2)(\neg P_{2-}(\tau, \tau_2, b_2))$.*

Suppose that the discrete-event system is modeled by

$$
\begin{array}{lll}
\delta P_1(\tau, \tau_1, x) & \leftrightarrow \delta u(\tau) \wedge (x = a_1) \wedge (\tau = \tau_1 - 1) & \text{-- add } a_1 \\
& +(\tau = \tau_1) \wedge (x = a_1) \wedge P_{1-}(\tau, \tau_1, a_1) & \text{-- remove } a_1 \\
& +(\tau = \tau_1 - 1) \wedge (x = a_2) \wedge P_{1-}(\tau, \tau_1 - 1, a_1) & \text{-- add } a_2 \\
& +(\tau = \tau_1) \wedge (x = a_2) \wedge P_{1-}(\tau, \tau_1, a_2) & \text{-- remove } a_2 \\
& +(\tau = \tau_1 - 1) \wedge (x = a_3) \wedge P_{1-}(\tau, \tau_1 - 1, a_2) & \text{-- add } a_3 \\
& +(\tau = \tau_1) \wedge (x = a_3) \wedge P_{1-}(\tau, \tau_1, a_3) & \text{-- remove } a_3
\end{array}
$$

and

$$
\begin{array}{lll}
\delta P_2(\tau, \tau_2, x) & \leftrightarrow \delta c(\tau) \wedge (x = b_1) \wedge (\tau = \tau_2 - 2) & \text{-- add } b_1 \\
& +(\tau = \tau_2) \wedge (x = b_1) \wedge P_{2-}(\tau, \tau_2, b_1) & \text{-- remove } b_1 \\
& +(\tau = \tau_2 - 2) \wedge (x = b_2) \wedge P_{2-}(\tau, \tau_2 - 2, b_1) & \text{-- add } b_2 \\
& +(\tau = \tau_2) \wedge (x = b_2) \wedge P_{2-}(\tau, \tau_2, b_2) & \text{-- remove } b_2
\end{array}
$$

where the initial time is 0, and P_{1-} and P_{2-} are empty for $\tau \leq 0$. In each equation an input impulse sets off a chain of events.

For example, an impulse of δu at time τ causes the ordered pair $< \tau + 1, a_1 >$ to be added to P_{1-} (assuming it is not already present). The time $(\tau + 1)$ recorded in this *pair is the time when this pair is scheduled for removal. At the same time the ordered pair* $< \tau+2, a_2 >$ *is added to* $P_{1\_}$. *Then at time* $\tau+2$, *the pair* $< \tau+2, a_2 >$ *is removed, and the pair* $< \tau+3, a_3 >$ *is added. At time* $\tau+3$ *the pair* $< \tau+3, a_3 >$ *is removed. One can view* a_1, a_2, a_3 *as denoting three different activities which are carried out in sequence and each of which takes one second.*

The sentence ψ *says that activities* a_3 *and* b_2 *cannot both be in progress at the same time.*

Since we are assuming discrete-event behavior, $\phi_{\_}(\tau)$ *has a pseudo- derivative, and*

$$\delta\phi(\tau) = (\theta_{1\_}(\tau) \vee \theta_{2\_}(\tau)) + ((\theta_{1\_}(\tau) + \delta\theta_1(\tau)) \vee ((\theta_{2\_}(\tau) + \delta\theta_2(\tau))) \tag{52}$$

Further,

$$\delta\theta_1(\tau) = \theta_{1\_}(\tau) + (\forall\tau_1)\,(\delta P_1(\tau,\tau_1,a_3) \leftrightarrow P_{1\_}(\tau,\tau_1,a_3)) \tag{53}$$

$$\delta\theta_2(\tau) = \theta_{2\_}(\tau) + (\forall\tau_2)\,(\delta P_2(\tau,\tau_2,b_2) \leftrightarrow P_{2\_}(\tau,\tau_2,b_2)) \tag{54}$$

Substitution into $\delta\phi$ *yields*

$$\delta\phi(\tau) = \phi_{\_}(\tau) \quad + \quad \{(\forall\tau_1)\,(\delta P_1(\tau,\tau_1,a_3) \leftrightarrow P_{1\_}(\tau,\tau_1,a_3)) \vee \tag{55}$$

$$((\forall\tau_2)\,(\delta P_2(\tau,\tau_2,b_2) \leftrightarrow P_{2\_}(\tau,\tau_2,b_2))\} \tag{56}$$

Assuming that $\phi_{\_}(\tau)$ *is satisfied, then* $\delta\phi(\tau) \leftrightarrow \perp$ *is satisfied if and only if the subformula* $\{\cdots\}$ *is satisfied. There are three ways that* $\phi_{\_}(\tau)$ *can be satisfied: neither* a_3 *or* b_2 *present;* a_3 *present and* b_2 *not present;* b_2 *present and* a_3 *not present. The subformula* $\{\cdots\}$ *describes* $\phi_{\_}$ *immediately after time* τ, *that is, it is* $\phi_+(\tau)$. *If* $(\forall\tau_1)(\delta P_1(\tau,\tau_1,a_3) \leftrightarrow P_{\_}(\tau,\tau_1,a_3))$ *is satisfied, then* a_3 *is not present just after time* τ *because* $P_{1+}(\tau,\tau_1,a_3) \leftrightarrow P_{1\_}(\tau,\tau_1,a_3)+\delta P_1(\tau,\tau_1,a_3) = P_{1\_}(\tau,\tau_1,a_3)+P_{1\_}(\tau,\tau_1,a_3) \leftrightarrow \perp$. *Similarly, if* $(\forall\tau_2)(\delta P_2(\tau,\tau_2, b_2) \leftrightarrow P_{2\_}(\tau,\tau_2,b_2))$ *is satisfied,* b_2 *is not present just after* τ. *Consequently, the disjunction of these two formulas says that not both* a_3 *and* b_2 *are present.*

Next substitute in the definitions of δP_1 *and* δP_2 *from the pseudo- differential equations. This yields after minor rearrangement the condition*

$$\begin{aligned} &(\forall\tau_1)\,[\{(\tau = \tau_1 - 1) \wedge P_{1\_}(\tau,\tau_1 - 1, a_2)\} \leftrightarrow \{(\tau_1 \neq \tau) \wedge P_{1\_}(\tau,\tau_1,a_3)\}] \\ \vee\ &(\forall\tau_2)\,[\{(\tau = \tau_2 - 2) \wedge P_{2\_}(\tau,\tau_2 - 2, b_1)\} \leftrightarrow \{(\tau_2 \neq \tau) \wedge P_{2\_}(\tau,\tau_2,b_2)\}] \end{aligned} \tag{57}$$

But we see that we do not have, yet, an explicit condition on δc *or* δu, *and we need such a condition. Furthermore, it will not help to take the pseudo-derivative of the above formula. It, after all, is merely* $\phi_+(\tau)$, *and its pseudo-derivative is the same as that of* $\phi_{\_}(\tau)$. *Consequently, we need another approach, and one possibility is shown in the next example.*

Example 13 *This is a continuation of the previous example.*

To finish the previous example we construct a partial solution of the pseudo-differential equations. In particular, we analyze the quantities $P_{1\_}(\tau, \tau_1, a_3)$ and $P_{1\_}(\tau, \tau_1 - 1, a_2)$ that appear in condition (57). From the definition of δP_1 we obtain

$$\delta P_1(\tau, \tau_1, a_3) \leftrightarrow (\tau = \tau_1 - 1) \wedge P_{1\_}(\tau, \tau_1 - 1, a_2) + (\tau = \tau_1) \wedge P_{1\_}(\tau, \tau_1, a_3) \tag{58}$$

and

$$\delta P_1(\tau, \tau_1 - 1, a_2) \leftrightarrow (\tau = \tau_1 - 2) \wedge P_{1\_}(\tau, \tau_1 - 2, a_1) + (\tau = \tau_1 - 1) \wedge P_{1\_}(\tau, \tau_1 - 1, a_2) \tag{59}$$

Since $P_{1\_}(\tau, \tau_1 - 2, a_1)$ occurs on the right of the second equation, we also need

$$\delta P_1(\tau, \tau_1 - 2, a_1) \leftrightarrow (\tau = \tau_1 - 2) \wedge P_{1\_}(\tau, \tau_1 - 2, a_1) + \delta u(\tau) \wedge (\tau = \tau_1 - 3) \tag{60}$$

The above three equations form the following system

$$\begin{bmatrix} \delta P_1(\tau, \tau_1, a_3) \\ \delta P_1(\tau, \tau_1 - 1, a_2) \\ \delta P_1(\tau, \tau_1 - 2, a_1) \end{bmatrix} \leftrightarrow \begin{bmatrix} (\tau = \tau_1) & (\tau = \tau_1 - 1) & \perp \\ \perp & (\tau = \tau_1 - 1) & (\tau = \tau_1 - 2) \\ \perp & \perp & (\tau = \tau_1 - 2) \end{bmatrix} \begin{bmatrix} P_{1\_}(\tau, \tau_1, a_3) \\ P_{1\_}(\tau, \tau_1 - 1, a_2) \\ P_{1\_}(\tau, \tau_1 - 2, a_1) \end{bmatrix} + \begin{bmatrix} \perp \\ \perp \\ (\tau = \tau_1 - 3) \wedge \delta u(\tau) \end{bmatrix} \tag{61}$$

where the obvious matrix manipulations are intended. This is a particularly simple system of equations to solve. The solution is

$$\begin{aligned} \begin{bmatrix} P_{1\_}(\tau, \tau_1, a_3) \\ P_{1\_}(\tau, \tau_1 - 1, a_2) \\ P_{1\_}(\tau, \tau_1 - 2, a_1) \end{bmatrix} \leftrightarrow & ((\tau_1 - 3) < \tau \leq (\tau_1 - 2)) \wedge \begin{bmatrix} \perp \\ \perp \\ \delta u(\tau_1 - 3) \end{bmatrix} \\ & + ((\tau_1 - 2) < \tau \leq (\tau_1 - 1)) \wedge \begin{bmatrix} \perp \\ \delta u(\tau_1 - 3) \\ \perp \end{bmatrix} \\ & + ((\tau_1 - 1) < \tau \leq \tau_1) \wedge \begin{bmatrix} \delta u(\tau_1 - 3) \\ \perp \\ \perp \end{bmatrix} \end{aligned} \tag{62}$$

where empty initial conditions have been assumed. More details of this solution method are presented in a later example.

Similarly,

$$\begin{bmatrix} P_{2\_}(\tau, \tau_2, b_2) \\ P_{2\_}(\tau, \tau_2 - 1, b_1) \end{bmatrix} \leftrightarrow ((\tau_2 - 4) < \tau \le (\tau_2 - 2)) \wedge \begin{bmatrix} \perp \\ \delta c(\tau_2 - 4) \end{bmatrix}$$
$$+ ((\tau_2 - 2) < \tau \le \tau_2) \wedge \begin{bmatrix} \delta c(\tau_2 - 4) \\ \perp \end{bmatrix}$$

Substituting into condition (57) we get

$$\begin{aligned} & (\forall \tau_1)\,[\{(\tau = \tau_1 - 1) \wedge \delta u(\tau_1 - 3) + (\tau = \tau_1) \wedge \delta u(\tau_1 - 3)\} \\ & \leftrightarrow \{((\tau_1 - 1) < \tau \le \tau_1) \wedge \delta u(\tau_1 - 3)\}] \\ \vee\ & (\forall \tau_2)\,[\{(\tau = \tau_2 - 2) \wedge \delta c(\tau_2 - 4) + (\tau = \tau_2) \wedge \delta c(\tau_2 - 4)\} \\ & \leftrightarrow \{((\tau_2 - 2) < \tau \le \tau_2) \wedge \delta c(\tau_2 - 4)\}] \end{aligned}$$

This is equivalent to

$$\begin{aligned} & (\forall \tau_1)\,[((\tau - 3) \le \tau_1 < (\tau - 2)) \rightarrow \neg \delta u(\tau_1)] \\ \vee\ & (\forall \tau_2)\,[((\tau - 4) \le \tau_2 < (\tau - 2)) \rightarrow \neg \delta c(\tau_2)] \end{aligned}$$

The first sentence says that there is no impulse δu at an earlier time that could cause a_3 to be present at time τ. The second sentence says the same thing about b_2.

The last condition can be rearranged to yield

$$\left[\delta c(\tau) \rightarrow (\forall \tau')\left\{(\tau \le \tau' \le \tau + 1) \rightarrow \neg \delta u(\tau')\right\}\right] \tag{63}$$

that is, a controlled input δc at time τ requires that there be no uncontrolled input δu in the future interval $(\tau \le \tau' \le \tau + 1)$, but the latter condition means that δc be selected on the basis of the future of δu. Since the future of δu is unknown at τ, the only choice for δc is $(\forall \tau)(\neg \delta c(\tau))$. In principle, then, it is possible to control the system to satisfy the original ψ; however, it is probably not attractive to say that no controlled input is ever allowed.

The problem with the above system is, of course, that the delay of the uncontrolled inputs is 3 while that of the controlled inputs is 4. If this were reversed, we could have more interesting δc's.

17. Additional Examples and Techniques

The following examples illustrate various aspects of the modeling methodology.

Example 14 *The following is a simple example from a class of models that is quite tractable. In particular, this class contains large models that can be attacked effectively with the method presented here.*

Let $\sum$ contain the single pseudo-differential equation.

$$\delta P(\tau, x_1, x_2) \leftrightarrow \delta e_1(\tau, x_1, x_2) \wedge P\_(\tau, x_1, x_2) + \delta e_2(\tau, x_1, x_2) \wedge P\_(\tau, x_2, x_1) \tag{64}$$

where δe_1 and δe_2 are formulas with no occurrence of $P\_$ or δP.

If one takes the informal view that x_1 and x_2 are "indices" over some index set, then for a fixed x_1 and x_2 $\delta P(\tau, x_1, x_2)$ can be viewed as a Boolean-valued function of time. It depends on $P\_$ for the indices $< x_1, x_2 >$ and for the indices $< x_2, x_1 >$. Thus, we also need a pseudo-differential equation to keep track of $P\_$ at $< x_2, x_1 >$, that is, we need

$$\delta P(\tau, x_2, x_1) \leftrightarrow \delta e_2(\tau, x_2, x_1) \wedge P\_(\tau, x_1, x_2) + \delta e_1(\tau, x_2, x_1) \wedge P\_(\tau, x_2, x_1) \tag{65}$$

Combining these two equations in a matrix format yields

$$\begin{bmatrix} \delta P(\tau, x_1, x_2) \\ \delta P(\tau, x_2, x_1) \end{bmatrix} \leftrightarrow \begin{bmatrix} \delta e_1(\tau, x_1, x_2) & \delta e_2(\tau, x_1, x_2) \\ \delta e_2(\tau, x_2, x_1) & \delta e_1(\tau, x_2, x_1) \end{bmatrix} \begin{bmatrix} P\_(\tau, x_1, x_2) \\ P\_(\tau, x_2, x_1) \end{bmatrix} \tag{66}$$

This, then, characterizes the behavior of $P\_$ at $< x_1, x_2 >$ and $< x_2, x_1 >$, and it is very simple to solve. However, there may be an infinite number of different sets $\{x_1, x_2\}$, so it would appear that this approach yields an infinite number of Boolean systems. In fact, it does, but, fortunately, things can usually be simplified. We will discuss the simplification, but first we discuss the solution of one of the Boolean systems.

It follows from $P_+ \leftrightarrow P\_ + \delta P$ that

$$\begin{aligned} \begin{bmatrix} P_+(\tau, x_1, x_2) \\ P_+(\tau, x_2, x_1) \end{bmatrix} &\leftrightarrow \{\delta\rho_0 O + \delta\rho_1 W_1 + \delta\rho_2 W_2 + \delta\rho_3 W_3 + \delta\rho_4 W_4 \\ &+ \delta\rho_5 W_5 + neI\} \begin{bmatrix} P\_(\tau, x_1, x_2) \\ P\_(\tau, x_2, x_1) \end{bmatrix} \end{aligned} \tag{67}$$

where

$$\begin{aligned} \delta\rho_0 &= \delta e_1(\tau, x_1, x_2) \wedge \delta e_1(\tau, x_2, x_1) \\ \delta\rho_1 &= \delta e_1(\tau, x_1, x_2) \wedge \neg\delta e_1(\tau, x_2, x_1) \\ \delta\rho_2 &= \neg\delta e_1(\tau, x_1, x_2) \wedge \delta e_1(\tau, x_2, x_1) \\ \delta\rho_3 &= \delta e_2(\tau, x_1, x_2) \wedge \delta e_2(\tau, x_2, x_1) \\ \delta\rho_4 &= \delta e_2(\tau, x_1, x_2) \wedge \neg\delta e_2(\tau, x_2, x_1) \\ \delta\rho_5 &= \neg\delta e_2(\tau, x_1, x_2) \wedge \delta e_2(\tau, x_2, x_1) \\ ne &= \textit{none of the above} \end{aligned} \tag{68}$$

	$\delta\rho_0$	$\delta\rho_1$	$\delta\rho_2$	$\delta\rho_3$	$\delta\rho_4$	$\delta\rho_5$
I	O	W_1	W_2	W_3	W_4	W_5
W_1	O	W_1	O	W_4	W_4	O
W_2	O	W_1	W_2	W_5	O	W_5
W_3	O	W_5	W_4	I	W_2	W_1
W_4	O	O	W_4	W_1	O	W_1
W_5	O	W_5	O	W_2	W_2	O
O	O	O	O	O	O	O

Table 1: The finite-state automaton

and

$$O = \begin{bmatrix} \perp & \perp \\ \perp & \perp \end{bmatrix}, W_1 = \begin{bmatrix} \perp & \perp \\ \perp & \top \end{bmatrix}, W_2 = \begin{bmatrix} \top & \perp \\ \perp & \perp \end{bmatrix} \tag{69}$$

$$W_3 = \begin{bmatrix} \perp & \top \\ \top & \perp \end{bmatrix}, W_4 = \begin{bmatrix} \perp & \top \\ \perp & \perp \end{bmatrix}, W_5 = \begin{bmatrix} \perp & \perp \\ \top & \perp \end{bmatrix}, I = \begin{bmatrix} \top & \perp \\ \perp & \top \end{bmatrix} \tag{70}$$

So if the interval $\tau_2 - \tau_1$ contains a finite number of events, then

$$\begin{bmatrix} P\_(\tau_2, x_1, x_2) \\ P\_(\tau_2, x_1, x_2) \end{bmatrix} \leftrightarrow M \begin{bmatrix} P\_(\tau_1, x_1, x_2) \\ P\_(\tau_1, x_2, x_1) \end{bmatrix} \tag{71}$$

where M is a product of the above matrices, and the product is determined by the sequence of the events $\delta\rho$. Initially M=I. After the first event M will be either $O, W_1, W_2, W_2, W_3, W_4,$ or W_5, depending on which $\delta\rho$ occurred. It happens that this set of matrices is closed under matrix multiplication, so M is always one of them, and one can model the progression of M's with a finite-state machine described in Table 1.

Thus we see that

- *There is a finite-state machine associated with each pair $< x_1, x_2 >, < x_2, x_1 >$.*
- *Since each machine has seven states, the set of all finite strings of $\delta\rho$ events associated with $< x_1, x_2 >, < x_2, x_1 >$ is partitioned into seven languages $L_o(x_1, x_2), L_1(x_1, x_2), \ldots, L_5(x_1, x_2), L_I(x_1, x_2)$*

The matrix M is determined by the language that the string of events up to the current time belongs to.

It turns out that we can represent each of these languages with a second-order formula, and if the language is star free we get a first-order formula.

Definition 6 *A language $W \subseteq A^*$, where A is an alphabet, is star free [12] if it can be generated from finite sets of strings by repeated application of the Boolean operations (union, intersection, and complementation) and concatenation.*

Below we show how to represent star free languages with first-order formulas. The languages are strings made up of events that occur between times τ_1 and τ_2.

Empty language:

$$L_\phi(\tau_1, \tau_2) = \textit{Some contradiction in } \delta\rho's \tag{72}$$

Language containing just the empty string:

$$L_\lambda(\tau_1, \tau_2) = (\forall\theta)\,[(\tau_1 \leq \theta < \tau_2) \rightarrow \textit{no event at } \theta] \tag{73}$$

Language containing just the string $\omega = \omega_1 ... \omega_n$:

$$\begin{aligned} L_\omega(\tau_1, \tau_2) \;\; &= \;\; (\exists! \theta_1 \theta_2 ... \theta_n)\,[(\tau_1 \leq \theta_1 < ... < \theta_n < \tau_2) \\ &\wedge \;\; \delta\rho(\theta_1) = \omega_1 \wedge ... \wedge \delta\rho(\theta_n) = \omega_n \wedge \textit{no other events}] \end{aligned} \tag{74}$$

Union:

$$L_1(\tau_1, \tau_2) \vee L_2(\tau_1, \tau_2) \tag{75}$$

Intersection:

$$L_1(\tau_1, \tau_2) \wedge L_2(\tau_1, \tau_2) \tag{76}$$

Complement:

$$\neg L(\tau_1, \tau_2) \wedge \textit{Discrete-event behavior} \tag{77}$$

Remark 34 *$\neg L(\tau_1, \tau_2)$ contains $\delta\rho's$ that do not have discrete-event behavior, and these have to be removed.*

Concatenation:

$$(\exists\theta) L_1(\tau_1, \theta) \wedge L_2(\theta, \tau_2) \tag{78}$$

Remark 35 *Needless to say, nonstandard models will exists for many of these languages.*

If the language is regular but not star free, we have to allow the star operation. It can be characterized with the following second-order formula

$$
\begin{aligned}
L^*(\tau_1,\tau_2) &= (\exists X)\,[X(\tau_1) \wedge X(\tau_2) \wedge (\forall\theta).(X(\theta) \rightarrow (\tau_1 \leq \theta \leq \tau_2)) \\
&\wedge \quad (X \textit{ has discrete-event behavior}) \wedge (N(\theta_1,\theta_2) \rightarrow L(\theta_1,\theta_2))]
\end{aligned} \tag{79}
$$

where $N(\theta_1, \theta_2)$ is the formula

$$
N(\theta_1,\theta_2) = X(\theta_1) \wedge X(\theta_2) \wedge \{(\forall\theta)\,((\theta_1 < \theta < \theta_2) \rightarrow \neg X(\theta))\} \tag{80}
$$

that is, θ_1 and θ_2 are neighboring points in X. The above formula is monadic second-order because of the existential quantification of the unary predicate variable symbol X. The purpose of X is to partition the interval $[\tau_1, \tau_2]$. The formula says that over each piece of this partition we have a string in $L(\theta_1, \theta_2)$.

If we denote the formulas for the seven languages in this example by $L_o(x_1, x_2, \tau, \tau_o)$, $L_1(x_1, x_2, \tau, \tau_o), \ldots, L_5(x_1, x_2, \tau, \tau_o), L_I(x_1, x_2, \tau, \tau_o)$, where τ_o is the initial time and is defined in terms of static quantities, for example, $\tau_0 = 0$, then we can represent the solution by

$$
\begin{aligned}
\begin{bmatrix} P_{\_}(\tau,x_1,x_2) \\ P_{\_}(\tau,x_2,x_1) \end{bmatrix} &\leftrightarrow \{L_o(x_1,x_2,\tau,\tau_o)O + L_1(x_1,x_2,\tau,\tau_o)W_1 + \ldots \\
&+ \quad L_5(x_1,x_2,\tau,\tau_o)W_5 + L_I(x_1,x_2,\tau,\tau_o)I\} \begin{bmatrix} P_{\_}(\tau_o,x_1,x_2) \\ P_{\_}(\tau_o,x_2,x_1) \end{bmatrix}
\end{aligned} \tag{81}
$$

If each of the seven languages is star free, the above is a first-order explicit solution. Otherwise, it is a second-order explicit solution. Note that even though we said that we were considering a fixed x_1 and x_2, we have, in fact, obtained a formula that works for arbitrary x_1 and x_2.

Finally, it is really more important that we found finite-state automata than that we found first and/or second-order formulas. The automata are usually the means by which we can do something practical.

Remark 36 *$\sum_\omega$ allows us to conclude that the above solution is unique.*

Although we have an explicit solution, there are some limits to its usefulness. In particular, there are still an infinite number of finite-state automata implicit in the explicit solution, and sometimes we cannot ignore this fact.

One way to go further is to assume some convenient form for δe_1 and δe_2. For example,

$$
\begin{aligned}
\delta e_1(\tau,x_1,x_2) &= \delta I_1(\tau) \wedge D_1(x_1,x_2) \\
\delta e_2(\tau,x_1,x_2) &= \delta I_2(\tau) \wedge D_2(x_1,x_2)
\end{aligned} \tag{82}
$$

occurs fairly often in practice. It follows that

$$\begin{aligned}
\delta\rho_o(\tau, x_1, x_2) &= \delta I_1(\tau) \wedge D_1(x_1, x_2) \wedge D_1(x_2, x_1) \\
\delta\rho_1(\tau, x_1, x_2) &= \delta I_1(\tau) \wedge D_1(x_1, x_2) \wedge \neg D_1(x_2, x_1) \\
\delta\rho_2(\tau, x_1, x_2) &= \delta I_1(\tau) \wedge \neg D_1(x_1, x_2) \wedge D_1(x_2, x_1) \\
\\
\delta\rho_3(\tau, x_1, x_2) &= \delta I_2(\tau) \wedge D_2(x_1, x_2) \wedge D_2(x_2, x_1) \\
\delta\rho_4(\tau, x_1, x_2) &= \delta I_2(\tau) \wedge D_2(x_1, x_2) \wedge \neg D_2(x_2, x_1) \\
\delta\rho_5(\tau, x_1, x_2) &= \delta I_2(\tau) \wedge \neg D_2(x_1, x_2) \wedge D_2(x_2, x_1) \\
\\
ne(\tau, x_1, x_2) &= \textit{no event.}
\end{aligned} \tag{83}$$

A simplification results because the D's partition the set of all possible $x_1, x_2's$ *into a finite set of blocks (at most 16), and one finite-state automaton can be assigned to each block. This means that at most 16 finite-state automata will characterize a solution. The common inputs to all automata are* δI_1 *and* δI_2*.*

Example 15 Kitting Station: *Consider a kitting station*[5] *at which three types of kits are assembled: the first has one part of type A, the second has one part of type B, and the third has a part of type A and one of type B. Custom pallets for the kits arrive at the station on a conveyor one after another at irregular times, and there is a bound on the number of pallets that can be passing through the station at a time. There are sensors that sense arrival times and kit type. The conveyor is such that there is a minimum time interval, m, between pallet arrivals. The appropriate parts are placed on a pallet while it is passing through the station. It is assumed that loading parts takes zero time and that after the pallet arrives there is a* Δ*-second interval during which the pallet may be loaded. The pallet must be loaded strictly after the beginning and strictly before the end of this* Δ*-interval. There are two infinite capacity bins at the station, one contains A parts and the other B parts. New parts are added to these bins at irregular times, and the bins may become empty. If two parts must be loaded, both must be loaded simultaneously. If a pallet cannot be loaded because of a lack of parts, it passes through the station.*

The above is an infinite state system because the bins have infinite capacity, and the leaving times can be any positive real number.

The sorts in the model are

- *time*
- *parts*

[5] A station where parts are combined into kits.

- *kit types*
- *pallet load status for parts of type A*
- *pallet load status for parts of type B*

The sentences Σ_T characterize time, and there are sentences defining Δ and m. There are three constant symbols of the sort kit type, k_A, k_B, k_{AB}, and there are sentences saying that these symbols denote all the kit types, and each denotes a different kit type. There are two constant symbols, p_A, a_A, of the sort pallet load status for parts of type A that are used to denote that a part of type A is "p"resent or "a"bsent from the pallet. Similarly, there are constant symbols p_B and a_B of the sort pallet load status for parts of type B.

Model for the conveyor-pallet subsystem:

In the dynamic predicate symbol family $\{C_-(\tau, y, \tau_l, u, v), \delta C(\tau, y, \tau_l, u, v)\}$, the predicate symbol C_- denotes pallets passing through the station. The symbol τ denotes the current time, y denotes the kit type, and τ_l denotes the time at which the pallet will leave the station. Since only one pallet leaves the station at a time, τ_l indirectly identifies a pallet. The variable symbols u and v denote whether or not parts of type A and B, respectively, are present or absent on the pallet.

The initial time is zero. We assume that C_- is constant-valued for all times less than or equal to zero; therefore, the history of C_- with respect to zero is determined by its value at time zero, $C_-(0, y, \tau_l, u, v)$. The intuitive idea is that this latter predicate is a set of pallets at the station at time zero. The predicate shows the type, leaving time, and load status for each of these initial pallets. The leaving time for one of these pallets must be nonnegative, so we require that $0 \leq \tau_l$ be satisfied. It takes Δ seconds to traverse the station, so these initial leaving times must satisfy $\tau_l < \Delta$, and equality is not allowed because that would correspond to a pallet arriving at time zero, and such pallets are not initial pallets. Thus, the following sentence must be satisfied.

$$(\forall \tau_l)(\forall y)(\forall u)(\forall v)[C_-(0, y, \tau_l, u, v) \rightarrow (0 \leq \tau_l < \Delta)]$$

This sentence is an initial condition.

The pallets are supposed to arrive unloaded at the station, but we allow the initial pallets to be loaded, that is, the pallets may have arrived and been loaded before time zero. Similarly, we allow any pallet type. Thus the following sentences must be satisfied.

$$(\forall \tau_l)(\forall u)(\forall v)[C_-(0, k_A, \tau_l, u, v) \rightarrow \{((u = p_A) \wedge (v = a_B)) \vee ((u = a_A) \wedge (v = a_B))\}]$$

$$(\forall \tau_l)(\forall u)(\forall v)[C_-(0, k_B, \tau_l, u, v) \rightarrow \{((u = a_A) \wedge (v = p_B)) \vee ((u = a_A) \wedge (v = a_B))\}]$$

$$(\forall \tau_l)(\forall u)(\forall v)[C_-(0, k_{AB}, \tau_l, u, v) \rightarrow \{((u = p_A) \wedge (v = p_B)) \vee ((u = a_A) \wedge (v = a_B))\}]$$

That is, we assume that an initial pallet is either empty or properly loaded. There are other initial conditions.

We restrict even further. Since there is a bound on the number of pallets that can be passing through the station at one time, we require that the difference in leaving times for two different initial pallets be greater than or equal to the known *minimum m. That is, the following sentence is satisfied.*

$$(\forall \tau_{l1})(\forall y_1)(\forall u_1)(\forall v_1)(\forall \tau_{l2})(\forall y_2)(\forall u_2)(\forall v_2)[C_-(0, y_1, \tau_{l1}, u_1, v_1) \wedge C_-(0, y_2, \tau_{l2}, u_2, v_2) \rightarrow$$

$$(((\tau_{l1} = \tau_{l2}) \wedge (y_1 = y_2) \wedge (u_1 = u_2) \wedge (v_1 = v_2)) \vee (m \leq |\tau_{l1} - \tau_{l2}|)]$$

That is, they are either the same pallet or their leaving times differ by at least m, that is, they are different pallets. This is another initial condition.

The following pseudo-differential equation models arrivals of pallets, their loading, and their departure. Each subformula is explained below.

$$\delta C(\tau, \overline{x}) \leftrightarrow$$

$$e_0(\tau, \overline{x}) + e_1(\tau, \overline{x}) \wedge D_1(\overline{x}) \wedge C_-(\tau, f_1(\overline{x})) + \cdots$$

$$+ e_4(\tau, \overline{x}) \wedge \{D_{41}(\overline{x}) \wedge C_-(\tau, f_{41}(\overline{x})) + D_{42}(\overline{x}) \wedge C_-(\tau, f_{42}(\overline{x})) + D_{43}(\overline{x}) \wedge C_-(\tau, f_{43}(\overline{x}))\}$$

where $\overline{x}$ is an abreviation for y, τ_l, u, v, the e_i's are subformulas characterizing events, and the f_i's and the f_{kl} are formulas characterizing vector-valued functions.

$$e_0(\tau, \overline{x}) = [\delta K(\tau, y) \wedge (\tau_l = \tau + \Delta) \wedge (u = a_A) \wedge (v = a_B)]$$

$$e_1(\tau, \overline{x}) = (\tau_l = \tau), \qquad D_1(\overline{x}) = T, f_1(\overline{x}) = (y, \tau_l, u, v)$$

that is, f_1 is the identity function.

$$e_2(\tau, \overline{x}) = [\delta A(\tau, \tau_l) \wedge (\neg \delta B(\tau, \tau_l)) \wedge (\tau < \tau_l < \tau + \Delta)] \; D_2(\overline{x}) = ((u = a_A) + (u = p_A)),$$

$$f_2(\overline{x}) = (y, \tau_l, a_A, v)$$

that is, f_2 maps u into a_A.

$$e_3(\tau, \overline{x}) = [\delta B(\tau, \tau_l) \wedge (\neg \delta A(\tau, \tau_l)) \wedge (\tau < \tau_l < \tau + \Delta)] \; D_3(\overline{x}) = ((v = a_B) + (v = p_B)),$$

$$f_3(\overline{x}) = (y, \tau_l, u, a_B)$$

$$e_4(\tau, \overline{x}) = [\delta A(\tau, \tau_l) \wedge \delta B(\tau, \tau_l) \wedge (\tau < \tau_l < \tau + \Delta)$$

$$D_{41}(\overline{x}) = [((u = p_A) \wedge (v = p_B)) + ((u = a_A) \wedge (v = a_B))], \qquad f_{41}(\overline{x}) = (y, \tau_l, a_A, a_B)$$

$$D_{42}(\overline{x}) = [(u = p_A) \wedge (v = p_B) + (u = a_A) \wedge (v = p_B)], \qquad f_{42}(\overline{x}) = (y, \tau_l, a_A, p_B)$$

$$D_{43}(\overline{x}) = [(u = p_A) \wedge (v = p_B) + (u = p_A) \wedge (v = a_B)], \qquad f_{43}(\overline{x}) = (y, \tau_l, p_A, a_B)$$

Consider the subformula $e_0(\tau, \overline{x}) = [\delta K(\tau, y) \wedge (\tau_l = \tau + \Delta) \wedge (u = a_A) \wedge (v = a_A)]$. The symbol $\delta K(\tau, y)$ denotes the arrivals of pallets at the station, τ the arrival time and y

the type. If $< \tau, y, \tau_l, u, v >$ is a tuple satisfying $e_0(\tau, \overline{x})$, then the value of the predicate C_- switches at time τ. Thus, if the value of C_- does not contain the tuple $< y, \tau_l, u, v >$ just before time τ, then it will just afterwards and vice versa.

δK is an exogenous quantity, and it is uncontrollable in the sense that it is not determined at the station. However, there are sentences which say that only one pallet arrives at a time and that arrivals are spaced by at least m.

Next consider the subformula $e_1(\tau, \overline{x}) \wedge C_-(\tau, f_1(\overline{x})) = [(\tau_l = \tau) \wedge C_-(\tau, y, \tau_l, u, v)]$. It says that whenever the current time, τ, equals the leaving time, τ_l, of a pallet, remove that pallet from the station. That is, at the end of the Δ-interval the pallet exits the station.

The subformula $e_2(\tau, \overline{x}) \wedge C_-(\tau, f_2(\overline{x})) = [\delta A(\tau, \tau_l) \wedge (\neg \delta B(\tau, \tau_l)) \wedge (\tau < \tau_l < \tau + \Delta) \wedge ((u = a_A) + (u = p_A)) \wedge C_-(\tau, y, \tau_l, a_A, v)]$ loads parts of type A onto a pallet. The load command is denoted by $\delta A(\tau, \tau_l)$, where τ denotes the time of the command and τ_l denotes the pallet, that is, the pallet with that leaving time. The $\neg \delta B$ says that a part of type B is not simultaneously being loaded. If there is no pallet with this leaving time, nothing happens. If there is, then there is a corresponding tuple in the predicate C_-, and if u in this tuple is not a_A, then nothing happens. Otherwise, a part of type A is loaded onto the pallet. δA denotes an exogenous quantity, but it is assumed to be controllable by the station, that is, a controller at the station can give this load command. Analogous remarks hold for the subformula $e_3(\tau, \overline{x}) \wedge D_3(\overline{x}) \wedge C_-(\tau, f_3(\overline{x})) = [\delta B(\tau, \tau_l) \wedge (\neg \delta A(\tau, \tau_l)) \wedge (\tau < \tau_l < \tau + \Delta) \wedge ((v = a_B) + (v = p_B)) \wedge C_-(\tau, y, \tau_l, u, a_B)]$

Note that the previous paragraph assumes that the bins are never empty. We will return to this assumption in a moment.

The subformula $e_4(\tau, \overline{x}) \wedge \{D_{41}(\overline{x}) \wedge C_-(\tau, f_{41}(\overline{x})) + D_{42}(\overline{x}) \wedge C_-(\tau, f_{42}(\overline{x})) + D_{43}(\overline{x}) \wedge C_-(\tau, f_{43}(\overline{x}))\}$ is similar to the previous two. It handles simultaneous loading of both types of parts.

The e_i's are mutually exclusive.

There are few useful facts that are listed below. Their proofs are not presented.

The following sentence says that the initial condition on the identity and spacing of pallets is true for all times.

$$(\forall \tau)(\forall \tau_{l1})(\forall y_1)(\forall u_1)(\forall v_1)(\forall \tau_{l2})(\forall y_2)(\forall u_2)(\forall v_2)[C_-(\tau, y_1, \tau_{l1}, u_1, v_1) \wedge C_-(\tau, y_2, \tau_{l2}, u_2, v_2) \rightarrow$$
$$(((\tau_{l1} = \tau_{l2}) \wedge (y_1 = y_2) \wedge (u_1 = u_2) \wedge (v_1 = v_2)) \vee (m \leq |\tau_{l1} - \tau_{l2}|))]$$

The next sentence says that if there is a pallet present, then it arrived at the time denoted by $\tau_l - \Delta$ or it is one of the initial pallets.

$$(\forall \tau)(\forall y)(\forall \tau_l)[(\exists u)(\exists v) C_-(\tau, y, \tau_l, u, v) \leftrightarrow$$

$$\{(\delta K(\tau_l - \Delta, y) \wedge (\tau \leq \tau_l < \tau + \Delta)) \vee (C_-(0, y, \tau_l, u, v) \wedge (0 \leq \tau_l < \Delta))\}]$$

Bin Subsystems

We model a bin with the following pseudo-differential equation[6]

$$\delta B(\tau, n) \leftrightarrow \delta I(\tau) \wedge (\neg \delta R(\tau)) \wedge$$

$$\{[(n = 0) \wedge (B_-(\tau) = 0)] + [(n \neq 0) \wedge ((B_-(\tau) = n) + (B_-(\tau) = n - 1))]\}$$

$$+$$

$$\delta R(\tau) \wedge (\neg \delta I(\tau)) \wedge$$

$$\{[(n = 0) \wedge (B_-(\tau) = 1)] + [(n \neq 0) \wedge ((B_-(\tau) = n) + (B_-(\tau) = n + 1))]\}$$

Σ_S contains first-order sentences describing enough of the natural numbers to allow addition and subtraction, that is, $n - 1$ and $n + 1$ are defined. B_- is a function whose value is the current contents of the bin. δI denotes the insertion of a part into the bin, and δR denotes removal. Simultaneous impulses in δI and δR have no effect.

Suppose that $B_-(\tau) = 10$ and there is an impulse in δI. Then the value of the first subformula on the right will be the set $\{10, 11\}$, and that of the second subformula will be empty. The effect will be to delete 10 from and add 11 to B_-. That is, one part is added to the bin. Next suppose that $B_-(\tau) = 10$ and there is an impulse in δR. Then the value of the first subformula is empty, and that of the second is $\{9, 10\}$. The effect will be to remove a part from the bin. Suppose that $B_-(\tau) = 0$ and there is an impulse in δI. Then the value of the first subformula is $\{0, 1\}$, and this causes a part to be added. Similarly, if $B_-(\tau) = 0$ and there is an impulse in δR, the value of the right side is empty, so nothing happens. If $B_-(\tau) = 1$ and there is an impulse in δI, the value is $\{1, 2\}$, and a part is added. Finally, if $B_-(\tau) = 1$ and there is an impulse in δR, the value is $\{0, 1\}$, and the bin becomes empty.

The two pseudo-differential equations developed above are linear in the sense that their right sides have a linear form with respect to exclusive-or "+". This is useful because simple solution methods are available for some linear equations. We will discuss this below. Unfortunately, when we combine these two pseudo-differential equations the result is not linear.

Interconnection of the subsystems

The model for the interconnection of the two subsystems is

$$(\forall \tau)(\forall \tau_l)[\delta A(\tau, \tau_l) \leftrightarrow \delta A'(\tau, \tau_l) \wedge (B_A(\tau) \neq 0)]$$

[6]Note that the symbol "+" is used in two ways: + for exclusive or and + for addition of natural numbers.

where $\delta A'$ is a new predicate symbol, and

$$(\forall\tau)[\delta R_A(\tau) \leftrightarrow (\exists\tau_l)\delta A'(\tau,\tau_l)]$$

The first sentence in effect replaces δA by $\delta A' \wedge (B_A \neq 0)$ in the first pseudo-differential equation, and the presence of $(B_A(\tau) \neq 0)$ destroys the linearity of this equation. The new loading command $\delta A'$ causes loading only when the bin is not empty. The second sentence says that the new loading command and a removal command must occur simultaneously. A similar pair of sentences applies to B parts.

Note that if we assume that the bins are never empty (as we did while developing the first pseudo-differential equation), then we get a kind of linearization. In particular, δA and $\delta A'$ become equivalent, and the source of nonlinearity disappears.

17.1. A General Solution Method:

We present a general solution method for linear pseudo-differential equations of the following form:

$$\delta P(\tau,\overline{x}) \leftrightarrow e_0(\tau,\overline{x})+$$
$$e_1(\tau,\overline{x}) \wedge \{D_{11}(\overline{x}) \wedge P_-(\tau, f_{11}(\overline{x})) + \cdots + D_{1k_1}(\overline{x}) \wedge P_-(\tau, f_{1k_1}(\overline{x}))\} + \cdots$$
$$+e_n(\tau,\overline{x}) \wedge \{D_{n1}(\overline{x}) \wedge P_-(\tau, f_{n1}(\overline{x})) + \cdots + D_{nk_n}(\overline{x}) \wedge P_-(\tau, f_{nk_n}(\overline{x}))\}$$

where $\overline{x}$ is an abreviation for $x_1,\ldots,x_l$, the e_i's are formulas, and each of the f_{ij}, $i = 1,\ldots,n$, $j = 1,\ldots,k_i$, is a set of l formulas defining a mapping of l-tuples to l-tuples[7] These mappings are distinct.

The $e_i \wedge D_{ij}$'s represent events and are formulas with no occurrence of P_- or δP. The e_i's are assumed to have discrete-event behavior. We also assume that the e_i's are pairwise mutually exclusive in the sense that $(\forall\tau)(\forall\overline{x}_1)(\forall\overline{x}_2)[\neg(e_i(\tau,\overline{x}_1) \wedge e_j(\tau,\overline{x}_2))]$ is satisfied. Thus, events associated with different e_i's cannot occur simultaneously; however, events associate with a common e_i, for example, $e_1 \wedge D_{11}$ and $e_1 \wedge D_{12}$, can be simultaneous.

Example 16 *The pseudo-differential equation for the dynamic function symbol family $\{C_-, \delta C\}$ in Example (15) is an example of such a pseudo-differential equation. The pseudo-differential equation for the dynamic function symbol family $\{B_-, \delta B\}$ is also in this form.*

Our solution method is based on creating a system of linear pseudo-differential equations. The basic idea is to add a new equation for each subformula of the form $P_-(\tau, f_{ij}(\overline{x}))$ appearing on the right side of the original pseudo-differential equation. However, the right sides of these new equations will contain subformulas such as $P_-(\tau, f_{ks}(f_{ij}(\overline{x})))$, and we need a pseudo-differential equation for the pseudo-derivative of this subformula. It can

[7]Since we are allowing a many-sorted language, these mappings will have to respect the sort of each argument of the predicate symbols P_- and δP.

happen that the composition $f_{ks}(f_{ij})$ is equal to a function for which we already have a pseudo-differential equation. If it is not, then another equation will have to be added. It in turn may have subformulas on its right side which require yet more equations. If the process terminates, the result is a finite system of equations, and this is the case of practical interest.

Example 17 *This is a continuation of Example 15.*

For the family $\{C_-, \delta C\}$, we have

$$\begin{array}{llllll}
f_1 f_1 = f_1 & f_1 f_2 = f_2 & f_1 f_3 = f_3 & f_1 f_{41} = f_{41} & f_1 f_{42} = f_{42} & f_1 f_{43} = f_{43} \\
f_2 f_1 = f_2 & f_2 f_2 = f_2 & f_2 f_3 = f_{41} & f_2 f_{41} = f_{41} & f_2 f_{42} = f_{42} & f_2 f_{43} = f_{41} \\
f_3 f_1 = f_3 & f_3 f_2 = f_{41} & f_3 f_3 = f_3 & f_3 f_{41} = f_{41} & f_3 f_{42} = f_{41} & f_3 f_{43} = f_{43} \\
f_{41} f_1 = f_{41} & f_{41} f_2 = f_{41} & f_{41} f_3 = f_{41} & f_{41} f_{41} = f_{41} & f_{41} f_{42} = f_{41} & f_{41} f_{43} = f_{41} \\
f_{42} f_1 = f_{42} & f_{42} f_2 = f_{42} & f_{42} f_3 = f_{42} & f_{42} f_{41} = f_{42} & f_{42} f_{42} = f_{42} & f_{42} f_{43} = f_{42} \\
f_{43} f_1 = f_{43} & f_{43} f_2 = f_{43} & f_{43} f_3 = f_{43} & f_{43} f_{41} = f_{43} & f_{43} f_{42} = f_{43} & f_{43} f_{43} = f_{43}
\end{array}$$

where, recall, some of the f's have a single subscript in the equation for $\{C_-, \delta C\}$. The significant point is that in this example the f-functions are closed under composition. In other situations this may not be true. That is, certain compositions of f functions may not be equal to any f function, and new pseudo-differential equations will have to be added. But one still hopes that only a finite number of equations need be added; however, this also need not be the case. For example, it is not the case for the pseudo-differential equation for the family $\{B_-, \delta B\}$ because of the $n-1$ and $n+1$ terms, so an infinite number of equations would have to be added.

Let G be the set of all functions that can be characterized by compositions of the f_{ij}-functions. We assume that G is finite, and let $G = \{g_1, \ldots, g_N\}$. Our convention is that g_1 is the identity function. The f_{ij}'s will be in G.

The system of pseudo-differential equations will be

$$\delta P(\tau, g_p(\overline{x})) \leftrightarrow e_0(\tau, g_p(\overline{x})) +$$
$$e_1(\tau, g_p(\overline{x})) \wedge \{D_{11}(g_p(\overline{x})) \wedge P_-(\tau, f_{11}(g_p(\overline{x}))) + \cdots + D_{1n_1}(g_p(\overline{x})) \wedge P_-(\tau, f_{1n_1}(g_p(\overline{x})))\} + \cdots$$
$$+ e_n(\tau, g_p(\overline{x})) \wedge \{D_{n1}(g_p(\overline{x})) \wedge P_-(\tau, f_{n1}(g_p(\overline{x}))) + \cdots$$
$$+ D_{nk_n}(g_p(\overline{x})) \wedge P_-(\tau, f_{nk_n}(g_p(\overline{x})))\}$$

where $p = 1, \ldots, N$. By our assumption that G is finite, each of the compositions of a f_{ij} and a g_p in the above formula will be equivalent to some g function. In other words, we will have a closed system of equations.

Example 18 *In Example 15 this system of equations for the family $\{C_-, \delta C\}$ is*

$$\delta C_-(\tau, f) \leftrightarrow E_0(\tau, f) + \{e_1(\tau, \overline{x}) \wedge W_1 + e_2(\tau, \overline{x}) \wedge W_2 + e_3(\tau, \overline{x}) \wedge W_3 + e_4(\tau, \overline{x}) \wedge W_4\} \wedge C_-(\tau, f)$$

where

$$\delta C(\tau,f)=\begin{bmatrix}\delta C(\tau,f_1(\overline{x}))\\ \delta C(\tau,f_2(\overline{x}))\\ \delta C(\tau,f_3(\overline{x}))\\ \delta C(\tau,f_{41}(\overline{x}))\\ \delta C(\tau,f_{42}(\overline{x}))\\ \delta C(\tau,f_{43}(\overline{x}))\end{bmatrix}\qquad E_0(\tau,f)=\begin{bmatrix}e_0(\tau,f_1(\overline{x}))\\ e_0(\tau,f_2(\overline{x}))\\ e_0(\tau,f_3(\overline{x}))\\ e_0(\tau,f_{41}(\overline{x}))\\ \bot\\ \bot\end{bmatrix}\qquad W_1=\begin{bmatrix}\top&\bot&\bot&\bot&\bot&\bot\\ \bot&\top&\bot&\bot&\bot&\bot\\ \bot&\bot&\top&\bot&\bot&\bot\\ \bot&\bot&\bot&\top&\bot&\bot\\ \bot&\bot&\bot&\bot&\top&\bot\\ \bot&\bot&\bot&\bot&\bot&\top\end{bmatrix}$$

$$W_2=\begin{bmatrix}\bot&\top&\bot&\bot&\bot&\bot\\ \bot&\top&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\top&\bot&\bot\\ \bot&\bot&\bot&\top&\bot&\bot\\ \bot&\bot&\bot&\bot&\top&\bot\\ \bot&\bot&\bot&\top&\bot&\top\end{bmatrix}\qquad W_3=\begin{bmatrix}\bot&\bot&\top&\bot&\bot&\bot\\ \bot&\bot&\bot&\top&\bot&\bot\\ \bot&\bot&\top&\bot&\bot&\bot\\ \bot&\bot&\bot&\top&\bot&\bot\\ \bot&\bot&\bot&\bot&\top&\bot\\ \bot&\bot&\bot&\bot&\bot&\top\end{bmatrix}$$

$$W_4=\begin{bmatrix}\bot&\bot&\bot&D_{41}(f_1(\overline{x}))&D_{42}(f_1(\overline{x}))&D_{43}(f_1(\overline{x}))\\ \bot&\bot&\bot&D_{41}(f_2(\overline{x}))&D_{42}(f_2(\overline{x}))&\bot\\ \bot&\bot&\bot&D_{41}(f_3(\overline{x}))&\bot&D_{43}(f_3(\overline{x}))\\ \bot&\bot&\bot&\top&\bot&\bot\\ \bot&\bot&\bot&\bot&\top&\bot\\ \bot&\bot&\bot&\top&\bot&\top\end{bmatrix}\qquad C_-(\tau,f)=\begin{bmatrix}C_-(\tau,f_1(\overline{x}))\\ C_-(\tau,f_2(\overline{x}))\\ C_-(\tau,f_3(\overline{x}))\\ C_-(\tau,f_{41}(\overline{x}))\\ C_-(\tau,f_{42}(\overline{x}))\\ C_-(\tau,f_{43}(\overline{x}))\end{bmatrix}$$

Example 19 *Since $C_+ \leftrightarrow C_- + \delta C$, in Example 18 we get*

$$\begin{bmatrix}C_+(\tau,f_1(\overline{x}))\\ C_+(\tau,f_2(\overline{x}))\\ C_+(\tau,f_3(\overline{x}))\\ C_+(\tau,f_{41}(\overline{x}))\\ C_+(\tau,f_{42}(\overline{x}))\\ C_+(\tau,f_{43}(\overline{x}))\end{bmatrix}\leftrightarrow\begin{bmatrix}e_0(\tau,f_1(\overline{x}))\\ e_0(\tau,f_2(\overline{x}))\\ e_0(\tau,f_3(\overline{x}))\\ \bot\\ \bot\\ \bot\end{bmatrix}+\{e_1(\tau,\overline{x})\wedge\begin{bmatrix}\bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\end{bmatrix}$$

$$+e_2(\tau,\overline{x})\wedge\begin{bmatrix}\top&\top&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\top&\top&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\top&\bot&\bot\end{bmatrix}+e_3(\tau,\overline{x})\wedge\begin{bmatrix}\top&\bot&\top&\bot&\bot&\bot\\ \bot&\top&\bot&\top&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\end{bmatrix}$$

$$+e_4(\tau,\overline{x})\wedge\begin{bmatrix}\top&\bot&\bot&D_{41}(f_1(\overline{x}))&D_{42}(f_1(\overline{x}))&D_{43}(f_1(\overline{x}))\\ \bot&\top&\bot&D_{41}(f_2(\overline{x}))&D_{42}(f_2(\overline{x}))&\bot\\ \bot&\bot&\top&D_{41}(f_3(\overline{x}))&\bot&D_{43}(f_3(\overline{x}))\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\bot&\bot&\bot\\ \bot&\bot&\bot&\top&\bot&\bot\end{bmatrix}+ne(\tau,\overline{x})\wedge I\}\wedge\begin{bmatrix}C_-(\tau,f_1(\overline{x}))\\ C_-(\tau,f_2(\overline{x}))\\ C_-(\tau,f_3(\overline{x}))\\ C_-(\tau,f_{41}(\overline{x}))\\ C_-(\tau,f_{42}(\overline{x}))\\ C_-(\tau,f_{43}(\overline{x}))\end{bmatrix}$$

where $n_e(\tau,\overline{x})$ denotes no event.

The discrete-event operation of the system will then be characterized as a product of the above matrices. However, there will only be a finite number of such products, and there will be a corresponding finite-state automaton that specifies the product that applies between τ_0 and τ.

18. Final Remarks

Remark 37 *The modeling methodology presented here is extremely flexible in that it can model a vast range of discrete-event systems. Unfortunately, many of the resulting models will be intractable in that it will be difficult to use them to solve practical problems. This is analogous to ordinary differential equations: most of them are intractable, too. Still, models based on ordinary differential equations are of enormous importance. One focuses on special cases, special techniques, and approximations. This approach applies to pseudo-differential equations as well. For example, we showed that linear pseudo-differential equations are a useful concept. We presented a useful solution technique, and we presented an example of linearization of a system of nonlinear pseudo-differential equations. And there are other useful classes of equations that are tractable. The point is, then, that a general approach to all possible models is impractical but specific ones are.*

Remark 38 *Although the models of discrete-event systems based on finite-state systems do have general solution methods, they quickly become impractical because of state-set explosion. This means that techniques to handle models with enormous state sets are needed. But this just means that one has to consider special classes of models, special techniques, and approximation. In other words, the finite-state models really have to be approached in the same ad hoc manner as our models. Arguably, our models are better for ad hoc approaches in that they make more of the system's structure and operation visible. For example, it was trivial for us to treat some infinite state systems.*

Remark 39 *Temporal logic has also been used for modeling discrete-event systems. The typical approach considers a special class of discrete-event systems and a restricted class of logical questions about system operation. Then an analysis algorithm is presented for this combination of systems and questions.*

So far we have been able to model each class of discrete-event system that is modeled by temporal logic and adapt its analysis algorithm to our framework. Thus, one can think of our modeling methodology as including temporal logic approaches as special cases. However, this is a mixed blessing since temporal logic analysis algorithms are applicable to only small systems.

Typically, each version of temporal logic is shown to be complete in some sense. Here we know that we may not get completeness. That is, even if we have a complete description of initial conditions and inputs, there may be a first-order sentence α such that neither α

nor $\neg\alpha$ can be deduced. Roughly speaking, temporal logic avoids this by restricting the class of sentences that is considered and tailoring the logic to the temporal framework. Since we allow all possible first-order sentences and insist on remaining within standard first-order logic, we may miss completeness. However, anything that can be proved in temporal logic can be proved within our framework. The point is that we may be able to prove more.

Remark 40 *Rule-based models are another approach to modeling discrete-event systems. There one has rules of the form $\alpha \mapsto \beta$, where α and β are first-order sentences. The idea is that if α is currently satisfied and the rule "fires", then β will be satisfied afterwards. If α is not satisfied, the rule cannot fire. The firing of a rule can be thought of as an event. The trouble with rule-based models is that they are ambiguous. Suppose that $\mathcal{A}$ is a structure modeling the current state of a discrete-event system and that $\mathcal{A}$ satisfies α. Further, suppose that the rule $\alpha \mapsto \beta$ fires. The new state will be a structure $\mathcal{B}$ that satisfies β. Unfortunately, knowing that $\mathcal{B}$ satisfies β, is not enough information to specify $\mathcal{B}$, and this is the sense in which rule-based models are ambiguous. To remove this ambiguity one needs to select a structure from among all those that satisfy β. The usual approach is to argue that $\mathcal{B}$ should be as close to $\mathcal{A}$ as possible, where saying what "close to" means becomes the key issue.*

One can view our modeling system as an unambiguous alternative to rule-based models. In fact, it grew out of an earlier effort to use rule-based models [13].

References

[1] P. J. G. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proceedings of the IEEE*, vol. 77, no. 1, pp. 81–98, January 1989.

[2] P. Varaiya and A. B. Kurzhanski, editors, *Lecture Notes in Control and Information Sciences, Discrete Event Systems and Applications*, number 103, Springer-Verlag, August 1987.

[3] Y. C. Ho, editor, *Proceedings of the IEEE, Special Issue on Dynamics of Discrete Event Systems*, volume 77, January 1989.

[4] C. G. Cassandras and P. J. Ramadge, "Special section on discrete event systems," *IEEE Control Systems Magazine*, pp. 66–112, June 1990.

[5] H. Enderton, *A Mathematical Introduction to Logic*, Academic Press, 1972.

[6] C. C. Chang and H. J. Keisler, *Model Theory*, North-Holland, 1973.

[7] R. Scheuring and H. Wehlan, "On the design of discrete event dynamic systems by means of the boolean differential calculus," in *IFAC Symposium, Zurich (Design Methods of Control Systems*, pp. 4–6, September 1991.

[8] A. Pnueli, "Applications of temporal logic to the specification and verification of reactive systems: A survey of current trends," in *Lecture Notes in Computer Science, Current Trends in Concurrency*, J. W. de Bakker, W. P. de Roever, and G. Rozenberg, editors, number 224, pp. 510–584, Springer-Verlag, 1986.

[9] J. S. Ostroff, "A logic for real-time discrete event processes," *IEEE Control Systems Magazine*, pp. 95–102, June 1990.

[10] R. Alur, C. Courcoubetis, and D. Dill, "Model-checking for real-time systems," in *Proceedings Fifth Annual IEEE Symposium on Login in Computer Science*, pp. 414–425, June 1990.

[11] E. Harel, O. Lichtenstein, and A. Pnueli, "Explicit clock temporal logic," in *Proceedings Fifth Annual IEEE Symposium on Login in Computer Science*, pp. 402–413, June 1990.

[12] W. Thomas, "Automata on infinite objects," in *Handbook of Theoretical Computer Science*, J. van Leeuwen, editor, chapter 4, pp. 134–187, Elsevier Science Publishers B.V., 1990.

[13] A. Naylor and R. Volz, "Design of integrated manufacturing system control software," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-17, no. 6, pp. 881–897, November/December 1987.