Creating Evolution Scenarios for Hybrid Systems

Victor W. Marek
Department of Computer Science
University of Kentucky
Lexington, KY 40506-0046
marek@cs.engr.uky.edu

Walton Sumner II
Department of Internal Medicine
Washington University
St. Louis, MO
sumner@oster.wustl.edu

Mirosław Truszczyński
Department of Computer Science
University of Kentucky
Lexington, KY 40506-0046
mirek@cs.engr.uky.edu

Abstract

We discuss the evolution scenarios for hybrid systems described by families of parameters changing in time. The systems are in discrete *states* and the transitions between states are probabilistic. The need for such scenarios arises in applications. In particular, in medicine such scenarions can be viewed as patient histories and can be used in medical training and testing.

1. Introduction

In this paper we investigate the issue of generating evolution scenarios for hybrid systems. The systems we have in mind are defined by continuously changing parameters and discretely defined states. States are determined by ranges of parameter values. Each system has its age (time from inception) and at any time moment it is in one of the states. Consider a state s and age t. The problem of interest is to develop a technique to create a hybrid system which at age t is in state s. This technique should endow the system with a history consistent with state s and explaining how the system evolved into it. The history is specified by a family of functions describing the parameters of the system over time until moment t.

Examples of such systems abound. Living organisms (including humans) can be viewed as systems in medical conditions (diseases as well as normal conditions [SC1, SC2]). Values of parameters such as height, weight, blood pressure, pulse, levels of hormone, etc define health states. Humans also exist as economic entities described by such parameters as wealth, investment preferences, level of education, levels of spending, savings, etc. Other examples include physical systems, such as industrial plants, with continuously changing parameters: engine power, fuel consumption, speed, direction, external conditions, etc.) or complex computer and telephone networks (with throughput capacity, and several traffic characteristics). In all these examples, states are determined by combinations of parameters, and systems change over time from one state to another.

An additional important characteristic of these systems is that the changes from state to state can be affected by actions of external "supervisors": physicians, financial advisors, plant and computer network managers. Over time, systems evolve from state to state. The goal of the supervisor is to avoid undesirable states, that is, to maintain the system in its current state if it is desirable, or move it to another, more desirable one. For instance, if the current economic condition of an individual is inconsistent with projected economic needs, then the financial advisor must recognize this fact and propose a strategy to correct this state of affairs.

Thus, in general, a supervisor should recognize the state of the system and design corrective actions. The first of these two tasks is the problem of diagnosis, requiring probing the system for current and past values of its parameters. Some of these values may not be available or may be very difficult or costly to obtain. Knowledge of past values is often helpful. In medicine complete or partial knowledge of the history of an individual is often critical for the correct diagnosis and efficient management.

A similar problem occurs in a class of games in which

we create a "context" describing the past of the characters occurring in the game. This history can influence the evolution of the game and players must be able to probe the game in order to make meaningful decisions.

We will outline a technique to generate evolution scenarios for hybrid systems. Availability of such scenarios is important for training and testing of system supervisors, particularly when training on a real system is prohibitively expensive, dangerous, or impossible for other reasons. For instance, in medicine, *simulated* training cases can be used by medical students and residents, or practitioners preparing for board certification/recertification examinations [SMT95, CBX]. Similarly, it is important for supervisors of physical systems, computer networks, etc., to practice responses to different scenarios.

Currently, creating simulations is expensive since, usually, the process requires substantial involvement of experts. For instance, certification and recertification examinations in medicine consist of large number of items written and verified by experts. Moreover, items cannot be reused easily. Automated generation of scenarios from a formal description of a hybrid system will make simulated training and testing practical.

2. Underlying model of a probabilistically changing hybrid system

To describe hybrid systems [KN93, BLL95, KNR95], we use the language of probabilistic automata. As usual, we have a discrete set of *states*, say S. The alphabet of our system consists of *events*. When s is a state, with various events we associate a time-dependent probability distribution which is used to select the subsequent state as well as the duration of the transition.

Each state is described by values of *parameters*. These parameters change in time. For example, the health state of diabetes in humans is described by parameters such as blood sugar level, with value > 140. To describe states pertaining to automobiles, we might use such parameters as the speed, power, fuel consumption, battery output, alternator output, etc. Formally, a parameter is described by a family of *patterns* — functions over time. We need to use families of patterns because, in general, a typical behavior of a parameter in a given state is expressed as a *range* of admissible values. For instance the output of the battery of a normal automobile ranges from 11.5V to 12.5V. Thus any function within that range will be considered admissible.

Patterns may have *subpatterns* describing deviations from the basic pattern if the system is subject to some actions. For instance, the pattern of high glucose associated with diabetes may have subpatterns showing that the level falls down after administering insulin, and rises after meals.

Since states are determined by values of associated parameters, for each state we will need an algorithm, called a *generation method*. For a given state s and a time moment t, when a system entered state s, the generation method returns concrete patterns for the relevant parameters (functions over a time interval starting at t). For example, the generation method for diabetes with the age of onset 45, returns specific patterns for blood sugar level between 180 and 240, consistent with this state.

Each created scenario will start in a unique *initial* state. Thus, in each hybrid system, one of its states will be distinguished as initial. It will be denoted by I. The patterns defining this state provide default values for other states. Namely, if a parameter p is not relevant to the generation method for a state s, its values will be instantiated from the pattern for p for the initial state. For instance, when creating a state of increased fuel consumption for a car, the battery output values available for the initial state will be used (assuming that fuel consumption has no relation with battery output).

Probabilistic information on transitions between the states is represented by the transition matrix TM [SC2]. For every combination of states p (intuitively, *predecessor* state) and s (*successor* state), $TM(p, s)$ is a function in two integer variables u and t (denoting times), where $0 \leq u, t \leq N$ and N is a large integer denoting the maximum lifetime of the system. The value $TM(p, s)(u, t)$ describes probabilities specifying the likelihood of transition from the state p, which started at u, to the state s, starting at t. Let us note that transitions between some pairs of states are not possible. In such case, the corresponding function will be set to 0. (In practice, depending on the structure of the network, more efficient representations of the matrix TM are possible.)

The transition matrix determines statistical data pertaining to states, specifically *incidence*, *duration risk*, *precursor risk* and *successor risk* functions. These data describe the probability that the system exists in a state and probability that the system enters the state during the specified time interval. By the *incidence* of state s at time t we mean the probability that a system enters state s precisely at time t. The

following formula defines incidence for states that are not initial in terms of the matrix TM:

$$inc(s,t) = \begin{cases} \sum_{p \in S} \sum_{u=0}^{t-1} TM(p,s)(u,t) & \text{if } t > 0 \\ 0 & \text{otherwise} \end{cases}$$

For the initial state I we have:

$$inc(I,t) = \begin{cases} 1 & \text{if } t = 0 \\ 0 & \text{otherwise} \end{cases}$$

By *(backward) duration risk*, $bdr(p,u,s,t)$, we mean the conditional probability of u being the time when p started, given that s was entered directly from p at time t. In terms of the transition matrix we have:

$$bdr(p,u,s,t) = \frac{TM(p,s)(u,t)}{\sum_{z=0}^{t-1} TM(p,s)(z,t)}.$$

Here the variable z ranges through all the possible times of the transition from p to s and so *bdr* expresses the conditional probability of s starting exactly at time moment t while p starts earlier. Finally, the *precursor risk*, $pr(p,s,t)$, is the probability that state s was entered directly from p and started at time t. State p could start at any time prior to t. Formally,

$$pr(p,s,t) = \frac{\sum_{u=0}^{t-1} TM(p,s)(u,t)}{\sum_{q \in S} \sum_{u=0}^{t-1} TM(q,s)(u,t)}.$$

Hence, *pr* expresses the chance that p is the predecessor of s and starts at any admissible time.

These functions will be used in the process of generating a history of a hybrid system in a given state and time. To evolve such system we will need the *forward duration risk* and the *successor risk*, which can be defined similarly and expressed in terms of the transition matrix. These notions will not be discussed here.

In this paper we deal only with the problem of generating evolution scenarios. To develop a complete training system we also need to represent actions that supervisors take and the effects of those. This requires an extension of the model by the notion of a *course of action*. Courses of action will play the role of control variables. The transition matrix can be expanded to contain information about their effects. This will be a subject of future studies.

3. The problem of scenario generation

We will provide now a formal statement of the problem. Given a state s and time moment t, the goal is to create a description of an instance of a hybrid system that enters state s at time t. This involves selection of:

1. a sequence of states $s_0, \ldots, s_k$ that starts in the initial state ($s_0 = I$) and leads to s ($s_k = s$).
2. a sequence $0 = t_0, \ldots, t_k = t$ of time moments, which are interpreted as starting times for states $s_0, s_1, \ldots, s_k$, respectively,
3. patterns for parameters consistent with with the sequence of states and their starting times.

The necessary constraint for the sequences s_i and t_i is that for every $i = 0, \ldots, k-1$,

$$TM(s_i, s_{i+1})(t_i, t_{i+1}) \neq 0.$$

This ensures that all the transitions are possible.

4. An algorithm

We will now outline an algorithm to solve the problem to generate evolution scenarios, as described above. Our discussion assumes that the transition matrix TM is given. In many application areas this matrix may not be directly available and has to be approximated from other data. We will point to several possibilities of approximating TM below. The details will be given in the full version of the paper.

The input to our algorithm consists of a state s and time moment t. The computation of the history consists of two phases. In the first one, we build (backwards) a sequence of states s_i and their start times t_i. We start from the given state s, with start time t, and iteratively compute the predecessors and their start times until we reach the initial state (usually the initial state). The start time of the initial state is set to 0. In the phase two, we proceed forward and construct functions (patterns) describing the parameters so that at time t_i the system is precisely in state s_i.

We will now describe Phase 1 of the history generation algorithm. Given a state s and its start time t, the goal is to construct a sequence of its predecessors (beginning with the initial state) and their start times. To find the predecessor s' of s and its start time t', we consider the conditional probabilities $P(p,u|s,t)$. Here, $P(p,u|s,t)$ stands for the probability that the predecessor state is p and that it started at u, given that s is the successor state and that it started at t. We have the following formula for $P(p,u|s,t)$:

$$P(p,u|s,t) = \frac{TM(p,u,s,t)}{\sum_{p' \in S} \sum_{u'=0}^{t-1} TM(p',u',s,t)}.$$

A pair (s',t') is selected according to this distribution. This process is then applied iteratively until the pair $(I,0)$ is reached.

In Phase 2, we define the parameters so that they are described by continuous functions consistent with the sequence of states produced earlier, and their start times. For instance, consider that the states are "increased fuel consumption" (defined as between 10 and 12 mpg) after 2 years of operating the vehicle, and "very large fuel consumption" (defined, say, as between 7 and 9.5 mpg) after 4 years. Then any function expressing mileage, having value between 10 and 12 mpg at year 2, between 7 and 9.5 mpg, at year 4, and gradually decreasing in the period between would constitute an appropriate description of the fuel consumption parameter. In the diabetes mellitus example, normal blood glucose levels of 50-60 mg/dl at birth would quickly rise to 80-110 mg/dl and persist until diabetes developed, when they would rise to 180-240 mg/dl.

The computational process of Phase 1 can be represented equivalently in terms of the precursor risk and duration risk. Specifically, we first use the precursor risks $pr(p, s, t)$, that is, the probabilities that state s was entered directly from p and started at time t, to select the predecessor state s' for s. Next, we use the duration risks $bdr(s', u, s, t)$, where $u = 0, 1, \ldots, t-1$, to select the start time t' for state s'. It is easy to see that

$$pr(p, s, t)bdr(p, u, s, t) = P(p, u|s, t).$$

Hence, the two approaches are indeed equivalent. However, the second approach may be easier to implement. The complete information about the transition matrix may be difficult or impossible to collect. This is the case in medical applications that we studied in which $TM(p, s)$ is not known. On the other hand, the precursor and duration risks are directly available or readily estimated.

Let us also notice that precursor and duration risks together with the incidence data allow us to reconstruct the transition matrix. Namely,

$$TM(p, s)(u, t) = pr(p, s, t)inc(s, t)bdr(p, u, s, t).$$

This implies an approach to determine or approximate the transition matrix: the incidence, the duration risk and the precursor risk must be collected or approximated. Subsequently, we can use the transition matrix, so computed, to determine the successor risk and forward duration risk necessary for simulating the evolution of the system. If the incidence, duration risk or precursor risk are approximated, then the quality of the resulting successor risk and forward duration risk may be insufficient for a faithful simulation. In such case, it may be better to approximate these functions directly.

5. Future work

The model presented in this paper assumes no additional structure on the notion of states. However, in many applications states are combinations of elementary states. Hence, it is possible to decompose the set S into the cartesian product of smaller and simpler sets of states $S_1, \ldots, S_k$. Each of these sets has its own network of transitions and its own transition matrix, and can be dealt with using the general techniques and algorithms presented here. Such approach results in a substantial reduction of the amount of data needed to be collected. However, the straightforward approach to determine the conglomerate transition matrix assumes independence of progression along different "parallel" networks. This is an unrealistic assumption in many applications where synergistic phenomena abound. For instance, in medicine a network modeling obesity will interact with a network modeling osteoarthritis. Similarly, a network modeling improper wheel alignment will interact with a network modeling the state of brakes. In our future work, we will develop means to model synergistic interactions in the context of several parallel networks.

The subject of this paper was generating histories of hybrid systems. However, the ultimate goal is to model control capabilities of hybrid systems. This will allow us to simulate system evolution and will have applications in training and education. Our current model will be expanded by the notion of an *event* or *course of action*. Such events alter probabilities of transition into a successor state. Hence, the transition matrix will have to have an additional parameter ranging over possible courses of action.

Finally, a specific research issue concerning the presented model is the robustness of approximations of the transition matrix. We will study the dependence of the quality of the transition matrix if computed from noisy data on incidence, precursor risk and duration risk.

References

[BLL95] A. Beneviste, B.C. Levy, and P. LeGuernic, A Calculus of Stochastic Systems. In: Hybrid Systems II, Springer Lecture Notes in Computer Science 999, pp. 21-44, 1995.

[CBX] Interim report on CBT phase II, National Board of Medical Examiners, February 1992, Philadelphia.

[KN93] W. Kohn and A. Nerode, Models for Hybrid Systems: Automata, Topologies, Controllability, Observability. In: Hybrid Systems, Springer Lecture Notes in Computer Science 736, pp. 317 – 356, 1993.

[KNR95] W. Kohn, A. Nerode, J.C. Remmel, and A. Yakhnis, Viability in Hybrid Systems, Theoretical Computer Science 138, pp. 141 — 168, 1995.

[SC1] W. Sumner, V.W. Marek, and M. Truszczynski, Designing a knowledge base to support family practice certification examinations. Proc. 17th Ann. Symp. Computer Applications in Medical Care 1993, p. 909.

[SC2] W. Sumner, V.W. Marek, and M. Truszczynski, Data Transformations for Patient Simulations. Proc. 19th Ann. Symp. Computer Applications in Medical Care 1995, p. 970.

[SMT95] W. Sumner, V.W. Marek, and M. Truszczynski, A Formal Model of Family Medicine, Journal of American Board of Family Practice 9(1996) pp. 41-52.